INTELLIGENT TECHNIQUES FOR DYNAMIC AND TRANSIENT ANALYSIS OF MULTI STAGE DESALINATION PLANT

by

Ayman Fahmy Abdelbary

This thesis submitted for the degree of

Doctor of Philosophy

at

Department of Electrical, Electronics and Information Engineering

City University

London

UK

April 1997

Thesis Summary

This thesis is concerned with dynamic and transient analysis of MSF desalination plants. The technique is developed using artificial neural networks (ANN) approach for the purpose of prediction, analysis, modelling, and control of MSF desalination plant. The applicability of the method to predict an approximation of the transient operating conditions as well as the control action are shown satisfactory. The network architecture and learning algorithm are developed based on the Multilayered Feed forward Networks (MFN) with the Back Propagation (BP) learning algorithm. It was shown that the approach could intelligently capture the dynamics of the system. An improved technique is developed for the BP learning algorithm based on *Global Error Node Evaluation* (*GENE*) approach for MFN to retains the function approximation requirements for a nonlinear dynamic behaviour. However, by using this approach considerable improvement for the generalization capability could be obtained for the case study under consideration. The technique provides the necessary dynamic learning behaviour required for MFN. This approach appears to be effective for the input - output dynamic modelling of complex process systems and therefore on-line adaptation is possible (when the characteristic of the system is changing or when more test data are available for another operating range). The developed algorithm is used for the development and validation of an empirical multi-controller structure for MSF desalination plant. Satisfactory results are obtained from practical examples with the additional training ability.

Summary of Original Contributions

To the best knowledge of the author, the following portion of the thesis is considered as the original contributions:

1. Improvement of the Back Propagation learning capability for Multi Layered Feed forward Networks using *GENE* approach.
2. Learning System Dynamics for MIMO using the MFN.
3. Applicability of using the Artificial Neural Network for assisting the modelling control problems found in MSF desalination process.

Part of the work described in this thesis is reported in the following publications:

1 Abdulbary A. F., L.L. Lai, D.M.K. Al-Gobaisi, "Application of Neural Network to Controlling and Modelling of a MSF Desalination Plant", Thirteen IASTED Int. Conf. MODELLING, IDENTIFICATION AND CONTROL, Feb., 21-23, 1994, Grindelwald, Switzerland, pp. 256-259

2 A F Abdelbary, L L Lai, D M K AlGobaisi and A Husain, "Experience of using the neural network approach for identification of MSF desalination plants" Desalination, 92 (1993) Elsevier Science Publishers B.V., Amsterdam, pp. 323-331

3 Abdulbary A. F., L L Lai, and D M K Al-Gobaisi, "Application of a connectioinest Model to controlling a MSF Desalination Plant", Proceeding of 3rd IEEE Conference On Control Applications, Aug. 1994, Glasgow, UK, Vol 3, pp. 1821-1822.

4 Abdulbary A. F, L L Lai, K.V. Reddy , A. Husain , and D M K Al-Gobaisi", Neural Networks as Efficient Tools of Simulation", Proceeding of IDA World Congress On Desalination and Water Reuse, Nov. 1995, Abu Dhabi, UAE, Vol IV pp. 361-374.

5 Abdulbary A. F., L L Lai, A. Husain , and D M K Al-Gobaisi, " An Introduction to Intelligent Control of Desalination Processes Using Neural Networks", Proceeding of IDA World Congress On Desalination and Water Reuse, Nov. 1995, Abu Dhabi, UAE, Vol I, pp 317.

ACKNOWLEDGMENTS

This research work was carried out in the Water and Electricity Department of Abu Dhabi. and at City University of London.

The author would like to thank his internal supervisor Dr. L. L. Lai for his help, valuable advice and guidance throughout the course of the research.

Thanks also to my external supervisor Dr. D M K Al- Gobaisi for his efforts and guidance during the course of the research, also to the efforts made to collect practical plant data. Also I would like to thank Dr. A Husain for his time and valuable discussion.

Finally, I would like to thank my father, wife and all my family for the encouragement and the support provided to complete this work.

Declaration

The author hereby grant powers of discretion to the City University to allow this thesis to be copied in whole or in part without further reference to the author. This permission covers only single copies made for study purposes, subject to normal conditions of acknowledgment.

Symbol Used

Symbol	Description
L	the number of layers of the network
l	the input layer
L	output layer
N_l	the number of network neurons in layer l
$x^l_j(k)$	the network input of the j^{th} network neuron of layer l at time k;
$S^l_j(k)$	the network output of the j^{th} network neuron of layer l at time k
$y_j(k)$	the network desired (target) output of the j^{th} network neuron at time k
$w^l_{ij}(k)$	the weight of the connection from i^{th} network neuron of layer l to j^{th} network neuron of layer $l+1$ at time k;
$g(.)$	the activation function of network neurons.
η	the learning rate
α	the momentum rate

Table of Content

List of Tables

List of Figures

Chapter 1 INTRODUCTION

1.1 Introduction

The desalination industry and related control system engineering have now developed in size and complexity to the point where increasingly sophisticated tools are necessary for solving the numerous problems that arise in operation, control, planning and design of these systems. This has seen the introduction of computer technology, the use of mathematical modelling and programming, control theory and simulation tools. Large classes of problems however, still continue to elude the above solution methodologies. These are frequently characterized by imprecision, heuristic decision making, non-linearity, large size and complexity.

In spite of the significant advances in linear control, the theoretical advances for non-linear control have been limited. Precise model of process dynamics becomes increasingly difficult to produce as process complexity increases. Precise modells are required in order to produce algebraic controller for such systems.

The desalination process is basically an evaporating and condensing process. The heat required for evaporation can be partly recovered during the condensing phase. MSF desalination is a type of evaporation with many technical and economic advantages and can be regarded presently as providing the optimum solution to the problem of sea water desalting. The basic principles and process description of the MSF desalination process are introduced later in chapter 2.

Like numerous other models, the model of the MSF is nonlinear in nature. This is mainly because of the dependence of physical properties of the streams upon temperature, pressure and salinity. In addition, relations for the heat transfer coefficients also contribute to the non-linearity of the model. All pervious works for modelling MSF process are based on the balance equations and the heat transfer equations in the linear form. The physical property functions that represent a complementary and basic part of the process model are main contributors to the complexity and non-linearity of the equations. They are the mathematical correlation expressions describing the thermo-physical properties of water, steam, and the brine solution. One important requirement of the correlation is their validity over a wide range of temperatures and concentrations. The evaluation of the overall heat transfer coefficient for the different stages and the brine heater is another factor of complexity. It is not possible to completely rely upon the use of mathematical models based on approximate empirical formula obtained through laboratory work.

Intelligent control attempts to solve difficult or complex problems by admitting process uncertainty or complexity and relaxing the specification on controlled response, rather than searching for an exact solution to simplified problems as in classical control. Control improvement can be achieved by design of a controller whose structure and consequent outputs in response to external commands are determined by experimental evidence, i.e., the observed input/output behavior of the plant, rather than by reference to mathematical or model-based description of the controller. The controller is then considered as intelligent controller in which the benefits are through decision making and/or learning capability. Both approaches can be combined. Dimitris *et al* (1992) shows that by combining the available prior knowledge of the system with the intelligent controller, further enhancement can be achieved where the intelligence will be learning and finding uncertainties and non-linearities of the system [1].

Advanced control techniques are intended to take into account all situations arising in real plant. Al Gobaisi *et al* (1991) analyzed the situation in some loops in an MSF plant and indicated the possibility of improvement by the application of advanced control technique [2]. The support of artificial intelligent (AI) to the application of advanced control strategies for the efficient operation of the MSF desalination plants are discussed in (AlGobaisi *et al*, 1994; Rao *et al*, 1994) [3], [4]. El-Hawary (1992) discussed possible application of artificial neural network (ANN) to desalination [5]. The main advantage of using such an approach is the provision of a general framework to tackle nonlinear control problems. Moreover, the engineering effort in developing a neural controller is less than for conventional controller design, at least for nonlinear systems.

In this thesis techniques using intelligent control through the use of ANN for dynamic modelling. However, model uncertainty, and control prediction are investigated and applied to the MSF desalination process. Benefits from using such techniques are discussed with emphasis to the required properties for *neurocontrol* application. One technique is based on creating an approximation of the operating conditions in time state space for specific conditions and network topology. This technique is presented to demonstrate the capability of *Multi-layered Feed forward Network* (MFN) for modelling and control of a MSF desalination plant based on practical examples available from the process. Comparative simulation results using data from the plant have shown that by adding more sensory information with proper choice of the learning rate and other learning parameters, the *neurocontroller* could generalize for all cases available when it is applied to the process under consideration. This is a very fast way for estimating controllers set points once the off-line training is successful. The weak point of ANN with feed forward configuration is the number of training examples required for generalization and the long training time. However, an improved technique based on *Global Error Node Evaluation* (*GENE*) approach, which retains the function approximation requirement for the back

propagation learning algorithm is developed. A novel network architecture and learning algorithm based on the back propagation using the *GENE* approach is shown to capture intelligently the dynamics of the system. The main benefit appears when more data are available for other conditions, the network can be further trained so that the previous and current learning are working for capturing the function approximation and not just interpolating the data. It is best suited to on-line approximate calculation. Moreover, MFN's with BP learning algorithm are characterized with the static behavior capability, however, by using this approach considerable improvement is obtained for the generalization capability of BP for MFN, as well as the technique provides the necessary dynamic behavior required for the MFN.

1.2 Out line of the thesis

Chapter 2: Overview of Desalination Process

The basic principles and process description of the MSF desalination process are introduced. Literature review on work done for modelling MSF desalination process including efforts for dynamic modelling are briefly introduced. Description of the control system required for MSF desalination process is discussed along with an application of a supervisory control system for MSF desalination process.

Chapter 3: Intelligent Control & Conventional Control System: An Overview

Mathematical modelling techniques have been traditionally popular for tasks such as the study of system behavior, process optimization, process control and the like, but owing to a variety of reasons like inadequate knowledge about the process, large computational cost of the attendant simulation etc., alternate strategies might be sometimes called for. Artificial neural networks (ANN) offer one such efficient and cost effective alternative.

Neural networks are essentially multidimensional regression paradigms based on input-output relationships. They are made to capture the knowledge about the characteristics of a system through a suitable learning process, so that a tuned network serves as an efficient tool for simulation studies. The present work attempts to highlight this aspect of neural networks with a case study conducted on a multistage flash (MSF) desalination plant.

This chapter reviews the literature in applying ANN to the control and modelling problem. The relationship of intelligent control to traditional control system is briefly reviewed. Next a comparison with the control system problems solving paradigm is carried out and lastly a brief overview of intelligent control methodology and application using the Artificial Neural Networks is discussed.

Chapter 4: Artificial Neural Networks (ANN)

Advances in both learning algorithms and microelectronics have renewed interest in neural networks across a spectrum of research areas. For control engineering, ANN are attractive because they have the ability of nonlinear plant modelling, can handle large amount of sensory information, perform collective processing and learning and offer the potential for highly parallel computation.

The most important characteristic of ANN is its ability to learn the frequently complex dynamic behavior of a physical system. Learning is the process where the network approximates the function mapping from system inputs to outputs, given a set of observations of its inputs and corresponding outputs. This is done by adjusting the network internal parameters, to minimize the squared error between the network outputs and the desired one. One such method is the error back-propagation (BP) algorithm, which is essentially a first order gradient decent method [6], and is discussed in detail. The results shown indicate the validity of the approach and gives some insight into the

behavior and generalization capability of the ANN. Problems concerning with convergence speed, the number of hidden neurons and the required number of training examples are discussed. A methodology is developed based on *global error node evaluation* (*GENE*) scheme for MFN, that account for generalization and avoiding local minima so that a consistent ANN approach can be developed for dynamic control of MSF. A new method based on *GENE* approach for MFN is developed. It retains the function approximation requirements for the back propagation (BP) learning algorithm for a nonlinear dynamic behavior. This approach appears to be effective for the input - output modelling of complex process systems and therefore on-line adaptation is possible (when the characteristic of the system is changing or when more test data are available for another operating range). Two problems in BP are addressed, namely, the saturation of the network nodes and the ultimate paralysis of the entire MFN during learning; and the problems of convergence to a local minima. In this approach the architecture is modified by adopting linear activation nodes at the output layer with fixed weights, while the hidden layers (two layers) are having nonlinear activation nodes. The *GENE* approach is validated using the relationship of the back propagation errors between each layer (output & hidden layers), and the subsequent weight update relation during the whole learning process.

By using multilayered feed forward network (MFN) for nonlinear empirical approximation for the model of interest, and adopting the static learning algorithm developed from the standard back propagation (BP) algorithm, it has shown to have the drawback of excessively long training time required for the development of even a simplified model of the process, hindering the development of a single model valid in its entire operating envelope. Alternatively, a *Global Error Node Evaluation* learning algorithm (*GENE*) for MFN is developed as an accelerated enhancement to train the MFN. The two main distinctions of this learning algorithm are: (1) no saturation for network nodes is exhibited avoiding the ultimate paralysis of the entire MFN during learning, (2) Global convergence

can be achieved by keeping the direction of the gradient in the same direction, avoiding the local minima problem. The internal behavior of the GENE hidden nodes is investigated and is shown to provide the necessary dynamic/nonlinear behavior that lacks in the standard BP for MFN.

Chapter 5: Application of Modelling & Control of MSF Plants by ANN-I

This chapter is devoted to demonstrate the capability of ANNs for modelling and control of a MSF desalination plant based on a description of practical examples from the desalination process and results of using ANNs for the simulation, identification, modelling, and control of MSF desalination plant. This is followed by a brief overview on the data used for the analysis and the various network structure used for learning. These include both transients and controlled variables that can be obtained from the various sensors. Next the ANN controller approach for set - point generation is described and lastly conclusion is drawn including possible direction of future research.

The approach used in this thesis is described. Data from the plant are used for training and testing. A Multi layer Feed forward Network (MFN) architecture with the BP learning algorithm is used and described. Control and modelling capability are investigated by considering the sample of the system input - output during load change following a time state space function V(t) with a mapping function N that can be approximated using ANN.

The set point estimation task using the ANN approach is investigated by using two ANN's; the first one (NNE) has the task of estimating the plant load/status trajectories during load variation, while the second one (NNC) has the task of producing the necessary control input based on NNE estimation. Thus the inputs to NNE network are selected as inputs and outputs for NNM. The outputs of the estimation network are which is the

subsequent input to another network. These outputs are the inputs to the control network (NNC) and the outputs are the manipulated variables (set point to regulatory control).

Chapter 6: Application of Modelling & Control of MSF Plants by ANN-II

To investigate the response of the control system to external disturbances such that the plant availability is not disturbed for MSF desalination plant, all main control loops were observed during imposed disturbances so that the interactions of the desalination process could be covered. The objective of this chapter is to present a dynamic black box model of the desalination process. The developed GENE algorithm is applied to data obtained from MSF desalination plant to identify and model the process behavior due to dynamic disturbances. The philosophy of applying ANN involve training on part of the scenarios obtained, and then, using the rest of the scenarios to test and study the network performance. The study includes the static test (slow variation of the process variables). The GENE approach is adopted as the learning algorithm, for which it is required to investigate what dynamic capability could be provided by the feed forward network. The internal behavior and convergence properties of the algorithm are compared to the standard back propagation with one hidden layer, and the advantages and disadvantages are discussed. The results are illustrated using data obtained from AL-TAWEELAH MSF plant at Abu Dhabi. Satisfactory results are obtained from simulation examples and they are given.

The performance of the network is studied by using dynamic test data from one control loop in a multi - controller structure for a large-scale process. Next it is discussed what kind of additional learning method is effective when new dynamic data are available from other control loops, so that the additional information provided are learnt efficiently

without affecting the previous learning process. The effective learning method is based on the Back Propagation (BP) algorithm with a variation or condition in order to: 1) it approximates an underlying mapping rather than interpolating training samples; 2) it is robust against gross error; 3) the rate of convergence is improved since the influence of incorrect samples is gracefully suppressed. It is important to examine the modelling capability of different ANNs, to determine what functional, representational and generalization properties they possess.

Chapter 7: Conclusion

Results for application of the ANN techniques using the *GENE* approach to predict the brine levels in the stages of MSF plants are summarized. Further work for the development of these techniques is discussed.

Chapter 2 OVERVIEW OF DESALINATION

2.1 Introduction

Vitality of water for life growth and lack of natural water sources in some areas of the world, especially in the Middle East and the Gulf areas, leads to sea water desalination for fresh water production. Different types of desalination plants are available, with the largest production capacity being in the conventional multistage flash desalination plants.

Generally the desalination process is by which pure water is separated from a solution of water and salts. In order to separate the pure water from the concentrated solution we have to supply some driving force for those separation. The provision of this driving force results into the energy consumption. Desalination processes can be split in two main categories:

1) Those where the separation involves a change of phase.
2) Those in which no change of phase is involved.

In the first category, energy is supplied to bring about a phase change between the product in the vapor form and the concentrated stream in the liquid form. It is important to note that, at the point of separation, there is no physical barrier between the two streams.

In the second category, the main processes are reverse osmosis and electrodialysis in which the product and concentrate streams are separated by a physical barrier called a membrane. The membrane allows the passage of only one material through it and so desalination is accomplished by the membrane allowing the passage of pure water through the membrane in case of reverse osmosis and by the passage of the charged ions through the membrane in the case of electrodialysis.

2.2 MSF Desalination Process

The desalination process in the first category is basically an evaporating and condensing process. The heat required for evaporation can be partly recovered during the condensing phase. MSF desalination is a type of evaporation with many technical and economic advantages and can be regarded presently as providing the optimum solution to the problem of sea water desalting.

The basic layout consists of a steam source, a water / steam circuit (brine heater) and an evaporator unit. The plant includes a brine heater and several flash stages divided into two sections - recovery and reject. The number of stages are determined according to the plant capacity and performance ratio. Optimum performance ratios are established on a case by case basis depending of fuel costs, quantity of distillate required, purpose and usage of the plant.

Steam is fed to the brine heater (to heat sea water) and ejector (to create vacuum). The circulating water is heated by the absorbed heat from the distillate and passes to the brine heater where the necessary heat is provided. The heated brine is flash evaporated in the

evaporator cells. The evaporator cells have condensers, through which relatively cool circulating water passes, the condensation takes place thereby producing distillate.

The stages are water sealed from each other to prevent pressure equalization between stages. Figure 2.1 shows the schematic process diagram of the multistage flash plant, and is limited to the main elements of the plant which are described in the as following section.

2.2.1 Basic Scheme of MSF Desalination Plant

The *cooling water,* which is chlorinated sea water from the sea water supply pumps, enters the plant as the "inlet heat reject flow". It flows through the condenser tubes of the heat reject stages on which vapor from the flashing brine condenses on the outside of the condenser tubes. Part of the cooling water heated in this way returns to the sea as heat reject flow. Only a fraction output of the flow is taken as *feed water* for the plant and is called the *make-up.* This feed is suitably conditioned and for this purpose a dosing unit and a deaerator are provided. The required make-up is added to the *recirculating brine* in the last stage. A combination of brine and make-up is extracted from the last stage of the evaporator by brine recirculating pump and is then pumped through the condensers of heat recovery section. The recirculating brine is treated with sodium sulphate to act as an oxygen scavenger and with antiscale to prevent scale before entering the heat recovery section. Here the vapor formed by the flashing brine is condensed. When the brine is recirculated through the heat recovery section condenser tubes, it absorbs the latent heat of the vapor formed by the flashing brine thereby raising its temperature and finally flows out from the condenser in the first stage. It then enters the *brine heater* and is heated by

steam to the required top brine temperature (TBT). Condensate formed in the brine heater returns to the condensate header.

From this point the brine flows into the bottom of the first chamber (flash *chamber).* Since it is superheated compared to the conditions in the flash chamber it flashes spontaneously and the brine temperature is lowered in accordance with equilibrium condition of the stage. The brine at its lower temperature flows through the inter-stage transfer orifices into the new stage in which the equilibrium temperature is a few degrees lower. Here, in all the subsequent stages, the flashing process is repeated, so that:

a) The brine cools down in the bottom of the chamber to the accompaniment of vapor formation.

b) The vapor formed by the flashing brine is condensed on the condenser tubes. The condensed water called distillate falls into the distillate tray, and

c) The brine in the condensers has its temperature raised. (refer to the "Temperature Profile" for an MSF desalination plant described below).

Through the flashing which takes place, the salt content of the brine increases stage by stage and reaches its maximum value in the final heat reject stage.

To prevent buildup of the salt content in the system, it is necessary to tap off a certain fraction of the brine at some points in the circuit and an equivalent amount of make up is taken as a feed. This make-up feed water is in addition to the feed water required to replace the distillate. This is done in the final heat reject stage at the point where the *blow down* flow is shown in the diagram, as this is the minimum temperature, reducing the heat

losses. Since this stage is under vacuum the flow diverted in this way must be extracted with a pump.

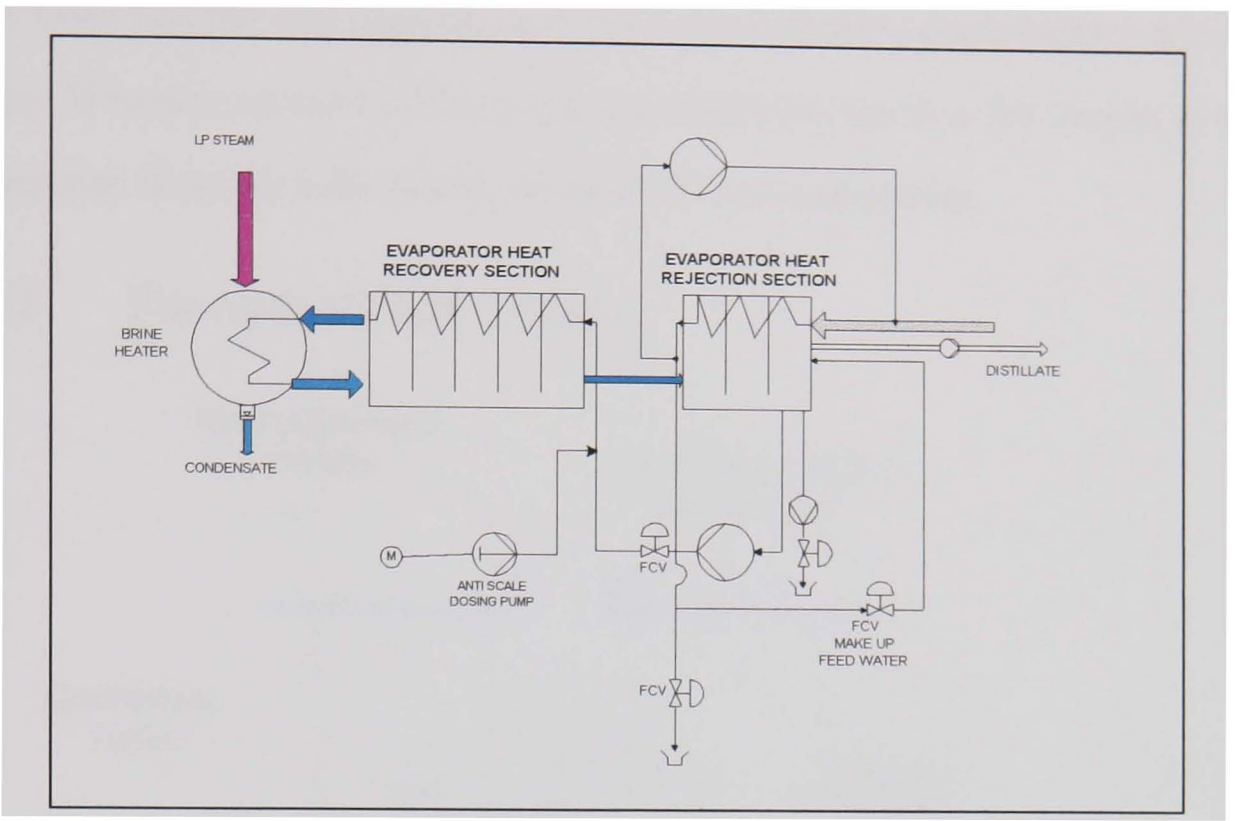


Figure 2.1: Schematic diagram of MSF plant

Within the stages the rising vapor passes through demisters before reaching the condensing tubes to prevent the distillate from being contaminated by any entrained sea water. The demisters usually consists of a closely woven wire mesh. The distillate falls from the condenser tubes and is collected in wide channels and flows through all the stages via special inter-stage transfer orifices. In flowing through the stages the distillate undergoes the same flashing process as the brine flowing underneath. It thereby follows that in each stage the sum of the distillate and brine flow between the inter-stages is constant and is equal to the value of the brine flow in the condenser tubes and before entry into the first flash chamber. *Distillate* must likewise be withdrawn by a pump from the final heat reject stage. *Vacuum* is usually created by connecting the flash chambers and the feed water deaerator to a steam jet ejector system. This ejector system will evacuate all air on start-up and also extract during the normal running of the evaporator all the non-

condensing gases, especially all the oxygen and carbon dioxide, which enter with the feed water and develop in the circuit as a result of thermal decomposition. In general each flash chamber vents into the next chamber with the non-condensing gases being removed at the last stage. However, as most of the gases are released in the first few stages, provision is made such that these are individually vented to the vacuum system.

2.2.2 Principal of Flash Chamber

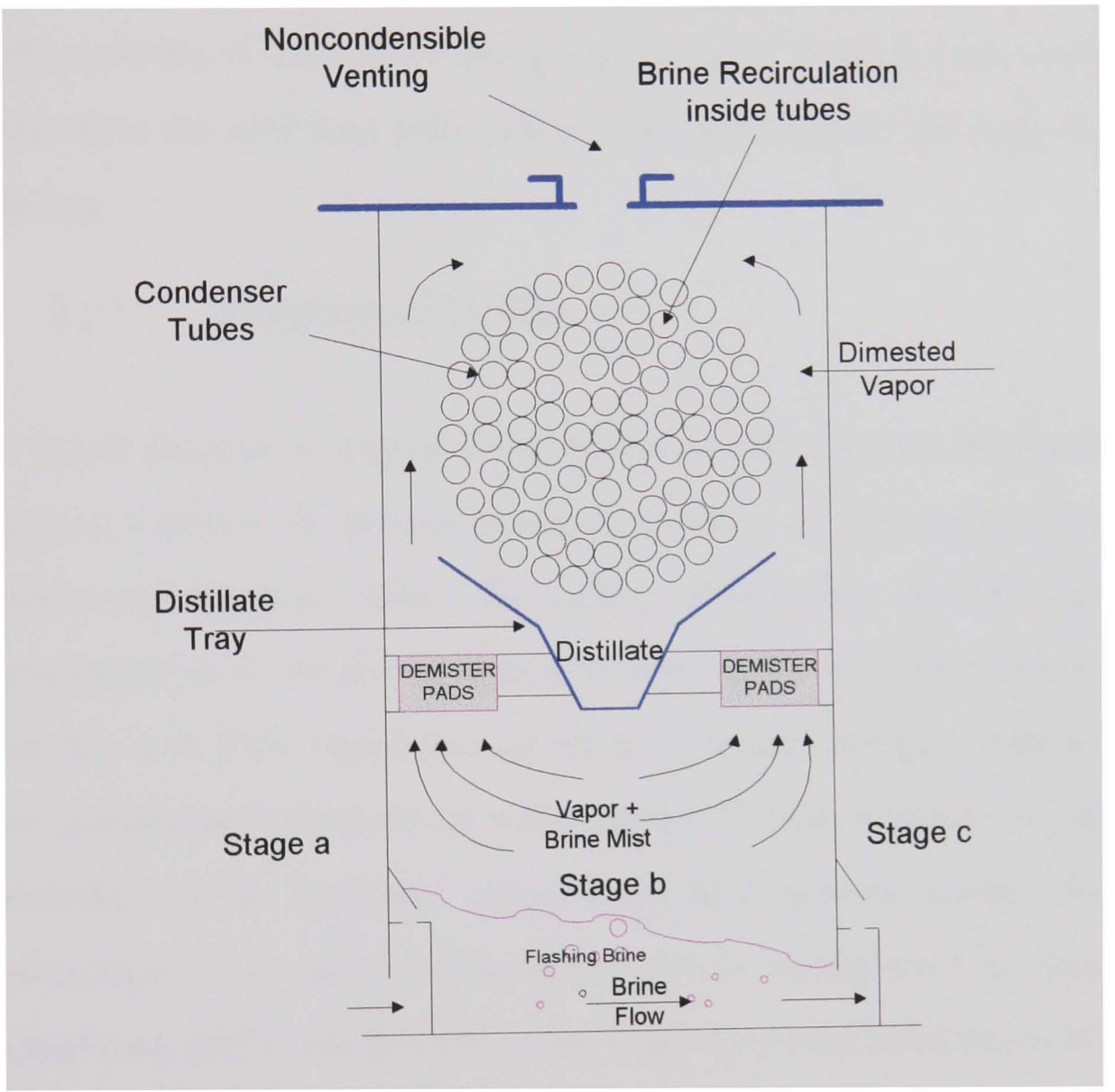


Figure 2.2: MSF stage (flash chamber)

The principle of flash chamber is illustrated in figure 2.2. If we take any stage of the evaporator the water vapor pressure prevailing in the flash chamber will be that corresponding to the equilibrium temperature reached in the stage. Hence the pressure in

the stage b, Pb is lower than that of the preceding stage Pa, but higher than that of the next stage Pc. *It is* this pressure difference which draws the brine through the evaporator stages. The brine enters a stage through the inter-stage brine orifices which are fitted with a brine gate which may be adjusted from inside the stage. This will enable the optimum brine level to be established in the stage during plant commissioning. A jump plate is fixed to the stage floor. This jump plate cause a wave to form thereby increasing the water depth in front of the brine orifice ensuring the inter-stage brine orifice is fully submerged and possibility of vapor blow through is eliminated. A splash plate is fixed to the stage wall above the inter-stage brine orifice to prevent the brine splashing, due to the sudden flashing

2.2.3 Temperature Profile

A typical temperature diagram for an MSF desalination plant is shown in figure 2.3. The diagram is divided into three areas corresponding to *the* heat reject section, heat recovery section and the brine heater. The cooling water *enters* the heat reject section at a temperature of t1. Within the condenser tubes of the heat reject section its temperature will rise to t4. This temperature and the brine temperature after flashing in the last stage are approximately identical, as with equality of these temperatures the vibrations are minimized. In the condenser tubes of the heat recovery section the brine flow (at temperature t4) is raised in temperature and on leaving the first stage has reached a temperature of t2 which also will be the brine heater inlet temperature. In the brine heater the temperature is increased to the maximum temperature in the circuit t3. This is the Top Brine Temperature (TBT). From the brine heater the brine flows at a temperature of t_3 into the first stage flash chamber where it is slightly cooled by the simultaneous flashing and vapor formation action. The flashing brine leaves the first stage at a slightly lower

temperature than that with which it entered and this change of temperature is approximately the same for all the stages. In the last stage the flashing brine drops to its lowest temperature within the circuit t_4. This is the Bottom Brine Temperature (BBT). The total brine temperature range of the plant i.e. the difference between the top brine temperature t_3 and the bottom brine temperature t_4 is called the "working range" or the "flashing range". The temperature of the distillate is lower than that of the flashing brine flowing beneath it by approximately 1 to 1.5 ^{o}C, i.e. by the amount of the boiling point elevation of the flashing brine and of various other temperature losses in the stage e.g. non-equilibrium, non-condensing gases, demister and bundle losses.

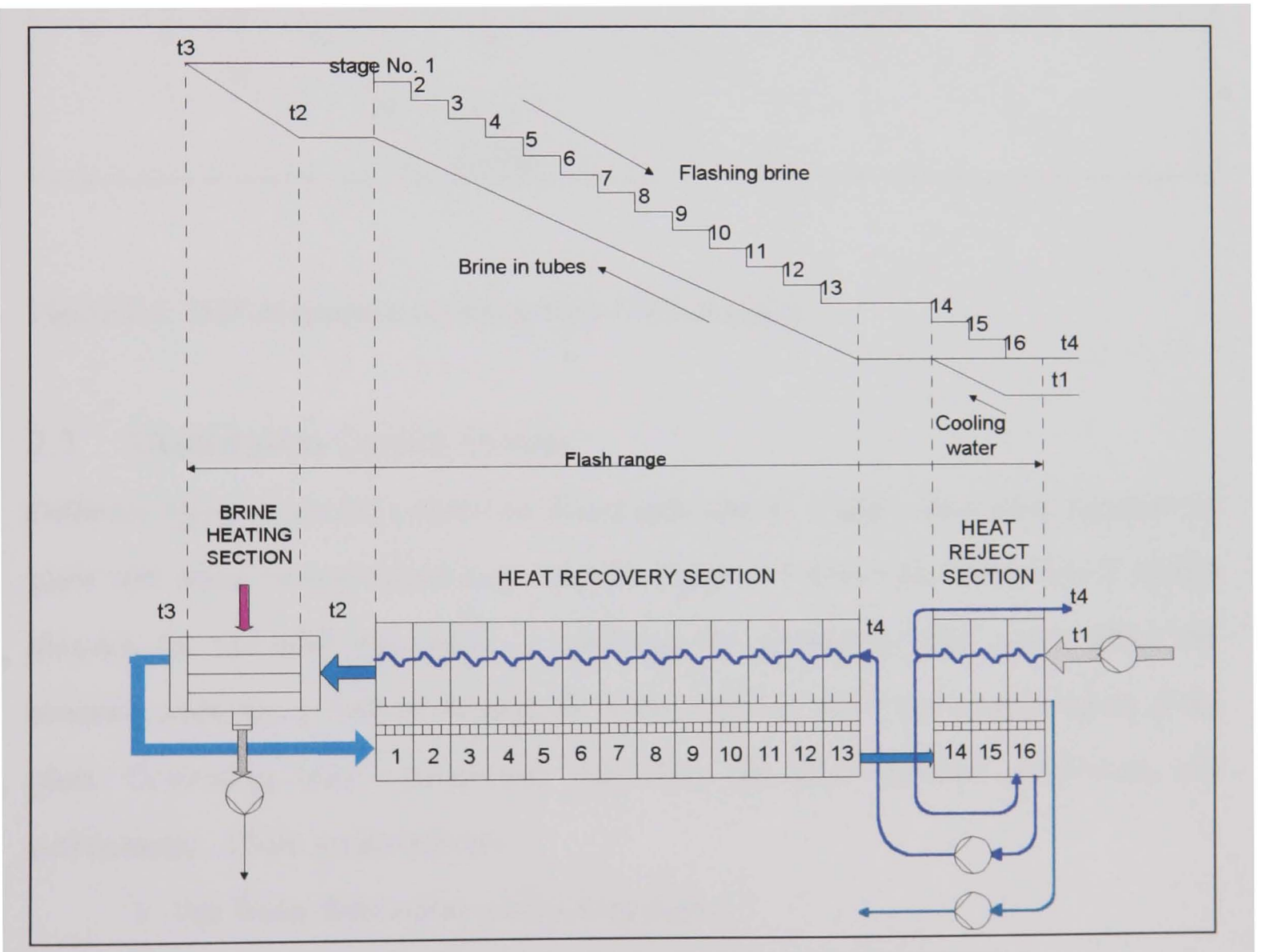


Figure 2.3 : MSF evaporator temperature diagram

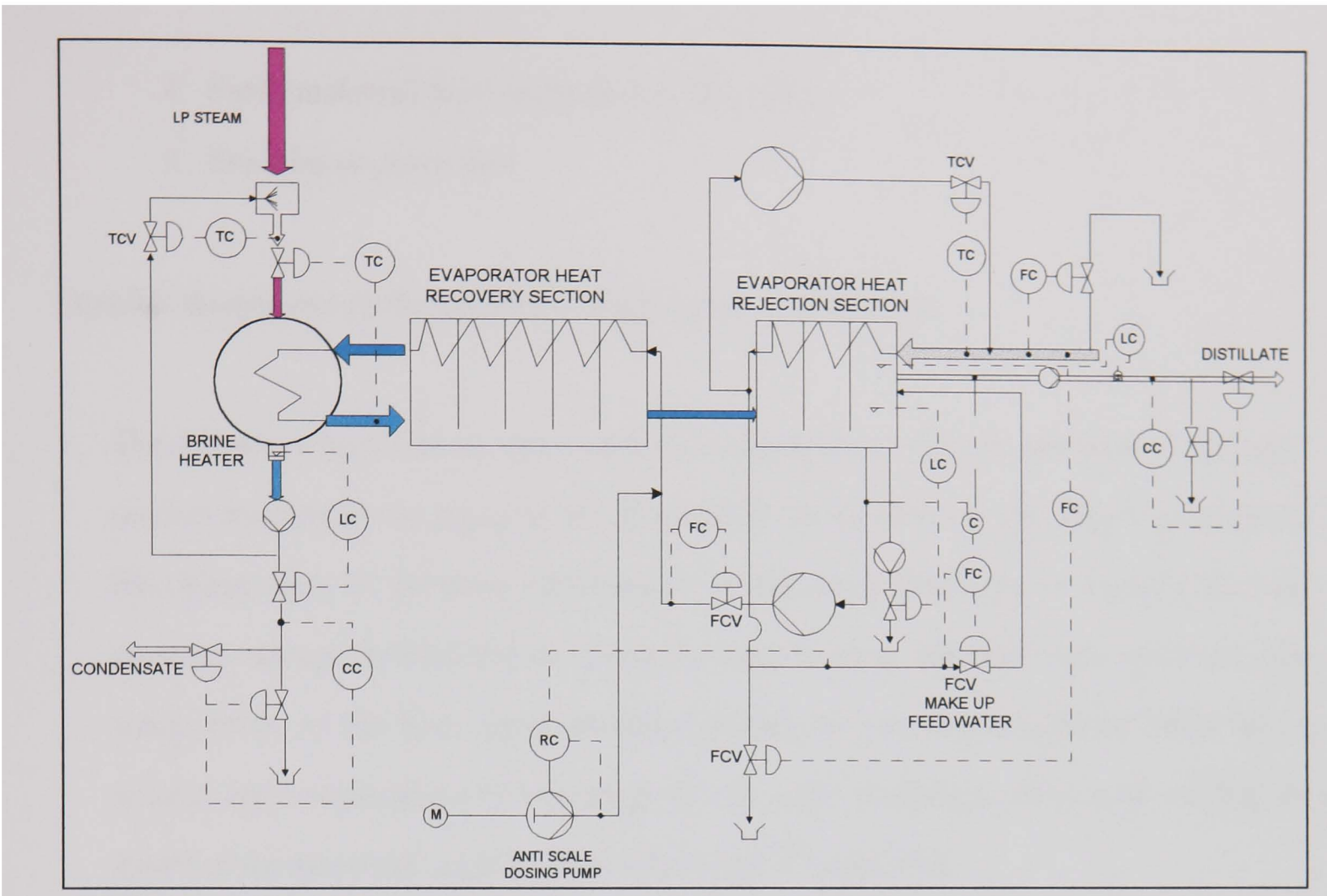


Figure 2.4: MSF evaporator control system / flow diagram

2.3 Desalination Control System

Different types of control system are found and used in a large scale MSF desalination plant with about twenty closed loop controls. Figure 2.4 shows the basic control system diagram for the MSF evaporator. Considering the parameters that would affect the process conditions, as well as the quantity of the *distillate* which is the end product of the plant. Controlling these parameters will affect the plant stability, production, and performance . These parameters are :

1. Sea Water from reject section temperature
2. Brine recirculation of flow rate
3. Brine from brine heater temperature (T.B.T)

4. Feed (makeup) flow rate/antiscale flow ratio
5. Brine blow down flow

Detailed description of the process control system is as follows:

1. The sea water flow rate to reject section is regulated so that the sea water from reject section temperature is equal to the temperature of the brine in the first stage which is the temperature of the brine recirculation at the pump discharge or entering the heat recovery section through the condenser to keep the heat balanced in the process. Sea water flows to the heat reject section depends on seasonal conditions (summer or winter) and is maintained by actuating the sea water discharge valve or by varying the speed of the sea water supply pump employing PI controller.

2. The brine recirculation flow is regulated to increase / decrease the production as it will increase/decrease the brine flashing rate. It will affect the quantity of brine that remains inside each stage. This must be done properly so that no disturbance occur in the levels of the brine in the different stages. The level in each stage should not be so high so that the carry over is encountered, which is the contamination of distillate product which result in higher salt content of the distillate than the required design level. At the same time the level should not be so low that will lead to blow through flow in the inter-stage. To maintain the brine flow rate, signals from electromagnetic flow transmitters are used to actuate the respective control valve. The objective is to maintain the brine recirculation and distillate flow rates at the required value in response to seasonal (summer, versus winter), and high temperature chemicals conditions affecting the flow rates.

3. The feed (makeup) flow rate is regulated so that the salinity of the recycled brine is controlled at a constant optimum concentration, this is achieved by a ratio control with the flow rate of the distillate drawn from the plant.

4. The control of distillate product flow is by regulating the Top brine temperature and the brine recirculation flow in reasonable sequences and at a rate of change that does not affect the equilibrium of the process. This rate is usually determined at site. The required brine temperature in order to produce a certain distillate product is regulated by variation of the steam temperature and flow rate to the brine heater. The steam temperature variation is achieved by variation of the de-superheating water flow. PID control is employed in the temperature controller. The steam pressure in the brine water is normally used as an auxiliary variable in the cascade control mode.

5. Brine level in the last stage chamber of the evaporator which is directly related to the levels in the preceding stages, is maintained at a predetermined set point. The PI level controller actuates the brine blow down valve. In the case of a brine blow down pump equipped with variable speed control, level control is done through speed change. For the system material balance it is required to keep the brine blow down flow rate equal to the difference between the makeup flow rate and the distillate output.

Additional loops such as the chemical injection loop, and sea water make up flow loop may be integrated in the overall control system. Further improvements are obtained when additional loops are added such as blow down flow, sea water recirculation, and heat reject section inlet temperature.

2.3.1 Process Control Requirement

Control of MSF desalination plants is generally based on conventional PID controller for TBT and only PI for the rest of the variables. However a derivative control function may be added if a fine control is desired. The controllers function fairly well when set properly at or near the best desired and calibrated set points. When a disturbance takes place, conventional controllers do not perform satisfactorily because the controller parameters settings do not correspond to the disturbance encountered.

MSF desalination plants with considerable amounts of mass and energy require novel types of high performance controllers that are almost always digitally based and perform control functions based on many available *modern* control algorithms.

Adaptive control (or self tuning control) offers one solution to the manual tuning of PID controllers, by attempting to maintain good robustness even if unpredictable changes occur in the process, sensors, and probably due to damage to the controller itself. The basic idea is to combine an on line parameter estimation procedure with some control system design technique to produce a control *law* with a self tuning capability. These controllers are capable of tuning themselves to optimal settings and returning whenever the process dynamics / behavior *changes*. The application of high performance *controllers* can result in a stable, efficient, smooth operating plant with extended life span. Adaptive control is successful only if a model exists with sufficiently accurate parameters. Model accuracy can be improved using artificial intelligence [3], [4].

For MSF desalination, control system design requires a knowledge of the inherent plant dynamic behavior, the really crucial flow-related control parameters are not those which are directly observable (such as temperature, flow, and pressure). Instead, they are the more indirect quantities such as overall (and localized) heat transfer coefficients in large

tube bundles, non-equilibrium losses in flashing of brine, and the stability of tray brine levels and inter-stage flows. Measurement of these phenomena must be derived from the direct measurements often requiring considerable accuracy. Proper modelling to understand the plant dynamic behavior is required and it would be great if the brine levels could be estimated accurately.

Like other processes, other modelling difficulties appears from the requirement for the inclusion of complex process characteristics such as time delays, disturbances, unmeasured variables, time-varying parameters, nonlinearities and multivariables interactions. While microcomputer based controllers are adequate hardware that is theoretically adequate to overcome these problems, the challenge is to find the appropriate software to direct the hardware. Artificial neural networks provide an innovative new paradigm that is beginning to be applied to these areas with excellent results.

2.4 The Dilemma of MSF Process Modelling

In general, the MSF process, as shown in figure 2.5, contains two recycle loops I, II. In the major recycle loop, part of the brine is recirculated to merge with the make-up flow. this combined stream, flowing counter currently to the flashing brine, is heated in the recovery section until it enters the brine heater. The second loop results by recycling part of the cooling water from the reject section to maintain constant cooling water temperature at the entry of the reject section (during the winter season).

The recovery and reject sections contain several stages. Each stage can be considered containing four compartments as shown in figure 2.6, namely, brine pool, product tray, vapor space and tube bundle; among these compartments the vapor and liquid flows are indicated by solid and broken lines, respectively.

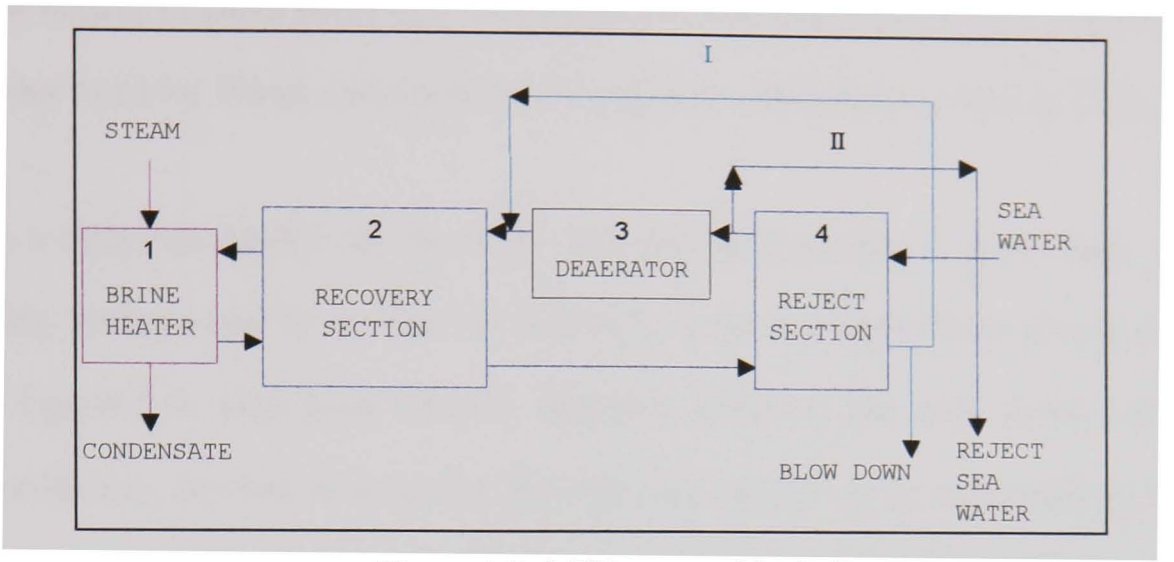


Figure 2.5: MSF process block diagram

Each fluid stream communicating with the individual stage has four attributes: Flow rate, temperature, pressure, and salt concentration, which can be considered as independent variables. For a detailed modelling the conservation of mass and energy for each stage must be satisfied, which are influenced by other stages. The model present a large scale problem where several hundreds of nonlinear equations must be solved. Various approaches were employed for solving such a model of nonlinear equations. One approach is based on the solution of these equations by stage to stage calculations, which is characterized by instability and low rate of conversions (e.g.: Beamer et al [7] used a stage to stage model with calculations being started at the hot end of the plant in an optimization study. A similar approach was used by Barba et al [8] where initial guess, provided by a simplified model which calculates the main dependent variables, were fed to the main model.). Another approach is to develop a rigorous method for solving the detailed steady state model which is based on the decomposition of the large set of equations into a smaller subsets followed by iterative sequential solution of these subsets. Rautenbach and Buchel [9], Omer [10] and Medani et al [11] followed a sequential approach with certain optimization program to minimize the stage wise fashion calculations starting at the hot end of the plant. Helal et al [12] linearized the governing equations for different sections of the plant. These were solved simultaneously using

tridiagonal matrix (TDM) technique originally proposed by Amundson and Pontinin [13] and was improved by Wang and Hanke [14] and was employed by Husain [15], [16].

To obtain a rigorous model for the MSF desalination process, a total mass, component and enthalpy balance can be written for each of the four compartment divisions of a stage shown in figure 2-6, plus heat transfer equation between the tube bundle and a vapor space. In addition, one has to consider that the evaporation process occurring in the brine pool is a constant enthalpy process. With time derivatives included, such a generalized model serves as dynamic model; when these derivatives are put equal to zero it represents steady-state condition [17]. This model has to be supported by accurate correlations for brine densities, boiling temperatures, brine and vapor enthalpies, and heat transfer coefficients. These parameters/coefficients which in the two-phase section is different from the parameters for one-phase section. They depend on the fraction of vapor, fluid velocity and the difference between pressure in the stage and the previous stage. In addition, there are temperature losses between the brine pool and the vapor space due to boiling point elevations, non-equilibration in the pool and pressure losses in the demisters and across the tube bundle, which must all be accounted in a representative model. Furthermore, there are uncertainties such as the fraction of the total evaporation which takes place from the product tray is not separately known. In a solvable analytical model along the above lines, simplifications lead to inaccuracies for which the main sources are the following :

- Heat transfer coefficients: the theoretical values differing 10% or more from those calculated from the actual plant operating data [7] may be due to the presence of non condensable gases. In addition, heat transfer rates decrease with time due to fouling. For on-line simulation, periodical measurements and calculations are necessary.
- During flashing if the thermal equilibrium is reached, then the temperature of the brine in each stage will be equal to the temperature of the vapor plus the boiling point

elevation. Normally, the equilibrium is not reached, therefore, there is an additional temperature loss that has to be accounted for in order to properly evaluate the vapor temperature.

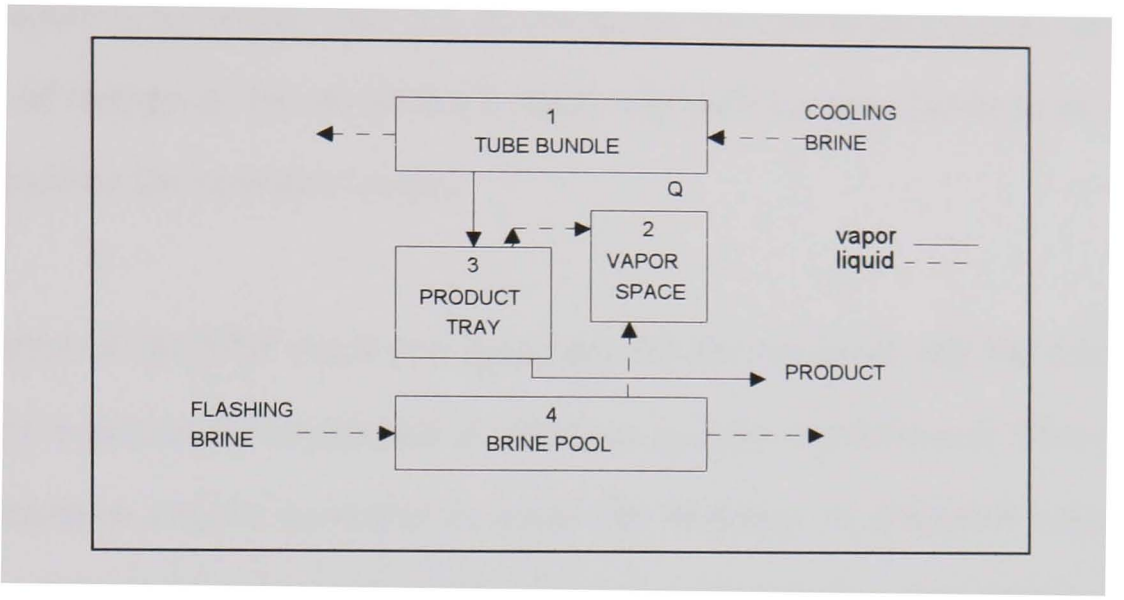


Figure 2.6: Representation of a single stage of the MSF plant

As regard the overall enthalpy balances, two approaches are possible, a local and a global approach. The balance is made with respect to terminal streams in each stage for the local approach. Whereas in the global approach, heat input to either recovery or reject sections is taken into account in making an enthalpy balance for each individual stage [8].

2.5 Supervisory Control System for MSF Desalination Process

One of the primary concern is to maintain the water production under all circumstances. MSF process startup requires the preparation of the vacuum into the stages and this requires some time and effort from the operator to create the vacuum required for the operation. For this plant shut down due to any disturbance is required to be avoided. Additionally load variation is frequently required in large MSF units. Hence a smooth, reliable and efficient control is required by the operator through the interface station for the various controllers. The change of distillate production in MSF plants is based on

controlling the heat input to the brine heater by decreasing / increasing the TBT controller set point in predetermined steps. For each step of the TBT change, the brine recirculation flow rate is also suitably changes in sequence. These steps are repeated until the required distillate product is achieved. The rate of change of the above parameters, determined by the amount of change in the set points in each step and the time between two successive steps, is judged by the operator on site.

Manual control of the MSF plant is a hard task for the operator and the dynamic model, considering the previously mentioned conditions, is quite complicated. Therefore, efforts have been taken to use the computer to assist the operator. A computer supervisory load variation control can be adopted and this has been realized at various locations in the Arabic Gulf such as Abu Dhabi and Oman [18], [19]. However, a dynamic model has not been used.

The values of the set points are calculated according to the followings:

- The brine recirculation flow rate is calculated using the operating curves as function of the product distillate flow rate and the cooling water temperature (reject inlet).
- The cooling water flow rate is function of the cooling water temperature (reject inlet)
- The TBT is calculated using the heat and mass balance of the evaporator and changes according with the changes in the fouling factor and in the sea water temperature.
- The steam temperature to the brine heater inlet is calculated according to the TBT. The calculation is performed in order to limit the degree of superheating of the steam at high load, and according to the maximum de-superheating water flow rate at low load.

The set point variation program consists mainly of two parts; the first is a successive step change command given to the controllers set points until the desired load is reached. The value of the step change is constant and is always determined by trial and error during the first commissioning of the plant. The second part is a waiting cycle, during which calculations and bound limit checks are usually performed. For example, the control deviations are checked on steady state behavior (e.g.: the control deviations have not exceeded 2% within the last 5 minutes). If the control deviations fulfill the steady state condition, set point variation steps can take place. Prior to executing the set point control program, a steady state check is made on the unit otherwise the program will not be executed. A steady state mathematical model is usually used to monitor the variations of the process parameters of the plant. A description of the mathematical model used and the equations involved in the calculation is reported in Appendix I [20]. During the execution of each step all external input and output modules are monitored. A faulty module will stop the program execution. Additionally the steam flow from the steam raising plant to the desalination unit is monitored in order to recognize if the supervisory control program over-charges the boiler capacity. The program will be switched back to latest step and stops if the steam flow exceeds 20 % of the value at the beginning of the step.

The foregoing supervisory control system is installed at Umm Al Nar East desalination plant in Abu Dhabi since 1988. During the first commissioning of the unit, and in order to obtain an efficient and reliable operation of the system, a continuous tuning of the coefficients and parameters was carried out for a period about three months. The main requirements are:

- to collect the operating fitting curves for the flow and temperatures,
- to calibrate the main parameters and
- to adapt the control logic to the desalination plant characteristics.

For example set point for TBT is calculated from a curve for load corresponding to TBT values. The curve parameters is tuned during the first commissioning of the unit. Further improvements to the procedure was based on observation and trial and error. Control logic adopted during a load variation for set points sequence of change is that at the start the change is affecting the TBT, l.p. steam to brine heater temperature and the antiscale dosing flow. After a time delay, the brine recirculation flow rate, the sea water to reject flow rate and the sea water to reject temperature start to be changed. The reason for this sequence of operations depends on the response of the process to the control actions and is explained in the following:

- The delay between the activations of the TBT and the brine recirculation flow rate is due to the need to develop the slope for the curve to change the temperature (increase or decrease), before any change of the brine recirculation affects it. In fact, an increase of the flow rate causes a decrease in the temperature, due to the difference response time of the process to the two changes of set points.
- The delay of the reject set points is due to the fact that their final values are calculated with a balance at the final running conditions, and a change in the reject temperature set point causes big fluctuations in the value of the sea water recirculation flow rate, because the time of change of the reject outlet temperature is very high.
- The set point of l.p. steam and antiscale dosing are functions of the TBT, so they are changed at the same time.
- The antiscale dosing (antiscale/make-up flow fraction) is a function of the TBT. The function depends on the kind of antiscale used. A lower limit of the dosing is fixed in order to avoid the alarm limit for the antiscale flow rate during normal operation at low load.

2.6 Conclusion

All pervious work for modelling MSF process are based on linearization of the balance equations and heat transfer equations. Validity of correlated values over a wide range of temperatures and concentrations is vital. Although several simulation results are reported in the literature, all are based on steady state simulation.

Steady state mathematical models for the MSF are available to be used for process steady state validation. The best model to be used does not require the user to be skilled model builder, and understand all programming and tuning details. Such that when further tuning is required the user can adapt the model to the process. However, an engineer who understands the process being simulated will not require more than a few days training in the neural net approach. In general, neural net approach has the advantage when the process data required to build the model are readily available. First principle methods are required to build models for operating regions not covered by the existing plant data, and are normally used with well-understood and previously modelled processes. The two approaches can be combined.

Neural net models can be quickly built and checked before implementation, typically taking less than one day. The required measurement tolerances can be statistically generated from information in the training data set. Further tuning is not necessary, as real plant data are used to build the models.

The key advantage of neural networks is that there is no need to build stage to stage or even short cut models of the desalination process from first principles. Neural net can build models describing any deterministic process relationship, without requiring the engineer to fully understand or describe the equations or the relationships between the process variables.

The key is to have a training data set that is large enough to contain information covering the full operating region of the plant. The model will then include influences of all inputs on the product quantity and quality. In the following chapters, the artificial neural network approach and its application to model the MSF desalination process is discussed in details and results from practical data obtained from the plant are introduced.

Chapter 3 Intelligent Control & Conventional Control System: An Overview

3.1 Introduction

Research and development for the control systems are continuing to keep abreast of the state of the art technology. Innovative concepts in control systems have relegated what was considered a satisfactory solution a decade ago to obsolescence today. One novel feature of *artificial intelligence (AI)* is its ability to handle the uncertainties and nonlinearities that we encounter in today's complex process control. One approach in the field of intelligent control is the *Artificial Neural networks (ANN)* controller or `neuro-controller'. System engineering in the desalination industry is used by a variety of different techniques to solve the modelling and control problem in this area. So the question that must be considered is how ANN would perform compared to other methods. In this chapter we address this issue. In the pervious chapter a discussion of a brief history of modelling and control problems that are found in the desalination technology. In this chapter a brief overview of the relationship of Intelligent Control to traditional control systems is discussed. Next a comparison with control system problems solving paradigm is carried out and lastly a brief overview of Intelligent Control methodology and application using the Artificial Neural Network is discussed.

3.2 Conventional and Intelligent Control

The term "conventional (or traditional) control" is used here to refer to the theories and methods that were developed in the past decades to control dynamic systems, the behavior of which is primarily described by differential and difference equations. Note that this mathematical framework may not be general enough in certain cases. In fact, it is well known that there are control problems that cannot be described in a differential/difference equations framework. Examples include discrete event manufacturing and communication systems, the study of which has lead to the use of automata and queuing theories in the control of systems.

In the minds of many people particularly outside the control area, the term "Intelligent Control" has come to mean some form of control using fuzzy and / or neural network methodologies. This perception has been reinforced by a number of articles and interviews mainly in the nonscientific literature. However, intelligent control does not restrict itself only to those methodologies. In fact, according to some definitions of intelligent controls, not all neural / fuzzy controllers would be considered intelligent. The fact is that there are problems of control which cannot be formulated and studied in the conventional differential / difference equation mathematical framework. To address these problems in a systematic way, a number of methods have been developed that are collectively known as intelligent control methodologies.

There are significant differences between conventional and intelligent control. It is worth remembering at this point that intelligent control uses conventional control methods to solve "lower level" control problems and that conventional control is included in the area of intelligent control. Intelligent control attempts to build upon and enhance the conventional control methodologies to solve new challenging control problems.

The word "control" in intelligent control has a different, more general meaning. than the word control in "conventional control". First, the processes of interest are more general and may be described, for example, by either discrete event system models, differential / difference equations models, or both. This has led to the development of theories for hybrid control systems that study the control of continuous-state dynamic processes by discrete-state sequential machines. In addition to the more general processes considered in intelligent control the control objectives also can be more general. for example, "fault tolerance techniques" can be the general task for the controller; this is then decomposed into a number of sub tasks, several of which may include, for instance, "follow a particular trajectory," which may be a problem that can be solved by conventional control methodologies. To attain such control goals for complex systems over a period of time, the controller has to cope with significant uncertainty that fixed feed back robust controllers or adaptive controllers cannot deal with. Since the goals are to be attained under large uncertainty, fault diagnosis and control reconfiguration, adaptation and learning are important consideration in intelligent controllers. So the control problem in intelligent control is an enhanced version of the problem in conventional control. It is much more ambitious and general. It is not surprising then that these increased control demands requires methods that are not typically used in conventional control. The area of intelligent control is in fact interdisciplinary, and it attempts to combine and extend theories and methods from areas such as control, computer science and operation research to attain demanding control goals in complex systems.

Note that the theories and methodologies from the areas of operations research and computer science cannot, in general, be used directly to solve control problems, as they were developed to address different needs. They must firstly be enhanced and new

methodologies need to be developed in combination with conventional control methodologies, before controllers for very complex dynamic systems can be designed in systematic ways. Also, traditional control concepts such as stability may have to be redefined when, for example the process to be control is described by discrete event system models. Rigorous mathematical frameworks, based for example on calculus, are being used to study such questions. However, in order to address control issues, these mathematical frameworks may not be convenient and they must be enhanced, or new ones must be developed, to appropriately address these problems. This is not surprising as the techniques from computer science and operations research are primarily analysis tools developed for non-dynamic systems, while in control, synthesis techniques to design real-time feedback control laws for dynamic systems are mainly of interest. In view of this discussion, it should be clear that intelligent control research, which is mainly driven by applications, has a very important and challenging theoretical component. Significant theoretical strides must be made to address the open questions and control theorists are invited to address these problems. The problems are nontrivial, but the payoff is high indeed.

Because intelligent control addresses more general control problems that also includes the problems addressed by conventional control, it is rather difficult to come up with meaningful benchmark examples. Intelligent control can address control problems that cannot be formulated in the language of conventional control. To illustrate this point, for example, in a desalination plant, while conventional controller may include flow regulators of the various streams, in the intelligent control framework one may include fault diagnosis and alarm systems. The problem of deciding on the set-point of the regulators, is based on the sequence of orders processed, selected, based on economic decisions, maintenance schedules, availability of machines etc.

Another difference between intelligent and conventional control is the separation between the controller and the system to be controlled. In conventional control the system to be controlled, called the plant, typically is separated from the controller. The controller is designed by the control designer, while the plant is in general given and cannot be changed; note that recent attempts to coordinate system design and control have been reported in areas such as space structures and chemical processes, since many times certain design changes lead to systems that are much easier to control. In intelligent control problems there may not be a clear separation of the plant and the controller; the control laws may be imbedded and be part of the system to be controlled. This opens new opportunities and challenges since it may be possible to affect the design of processes in a more systematic way.

3.3 A Comparison with Control System Problems Solving Paradigm

Solving the problem of any control design in system analysis starts with a description of the problem in the form of state space or a set of linear or non-linear algebraic equation to characterize the problem description. Then by using a numerical solution to find the 'stable' or 'optimal' solution to the problem is to be solved. This relies on being able to formulate the problem at hand into mathematical formalism. In contrast, neural networks allow one to deal with problems that cannot be formulated in this manner.

The neural network is essentially a model because it seeks to develop a model of the problem solver rather than the problem itself. The adaptation law in the neural network represents the manner in which the problem solver tackles the problem, rather than being a description of the problem. This opposite direction is motivated for the neural network

approach by the learning ability and the generalization capability, if guaranteed. As shown in Figure 3.1 a block diagram showing (a) the steps followed by the normal system analysis versus (b) neural network approach. We note that modelling the problem is embedded in modelling the problem solver to find a solution using the learning ability of the neural network.

System analysis approach uses the problem solver directly on the formulated model, while for the neural network approach it is used directly after solving the problem, and no further solution is searched under any condition. From here a restriction or bounds are added to the problem in order to be in the domain of the required generalization. This is implemented by including constrains which can be included during or after the learning phase. Alternatively is by using the first solution with the second one to include *a priori* information in hybrid solution. This requires enough information to be available to reach a solution. As the learning is performed by adjusting the weights in a supervised /unsupervised mode in order to find a solution for representing or solving the problem in the same time, i.e. finding a solution for the two problems.

For practical modelling application and to find an acceptance for the control community, researchers in this field have analyzed the problem to find a systematic approach to the problem solving. However, the acceptance criteria for neural network approach is in the generalization capability.

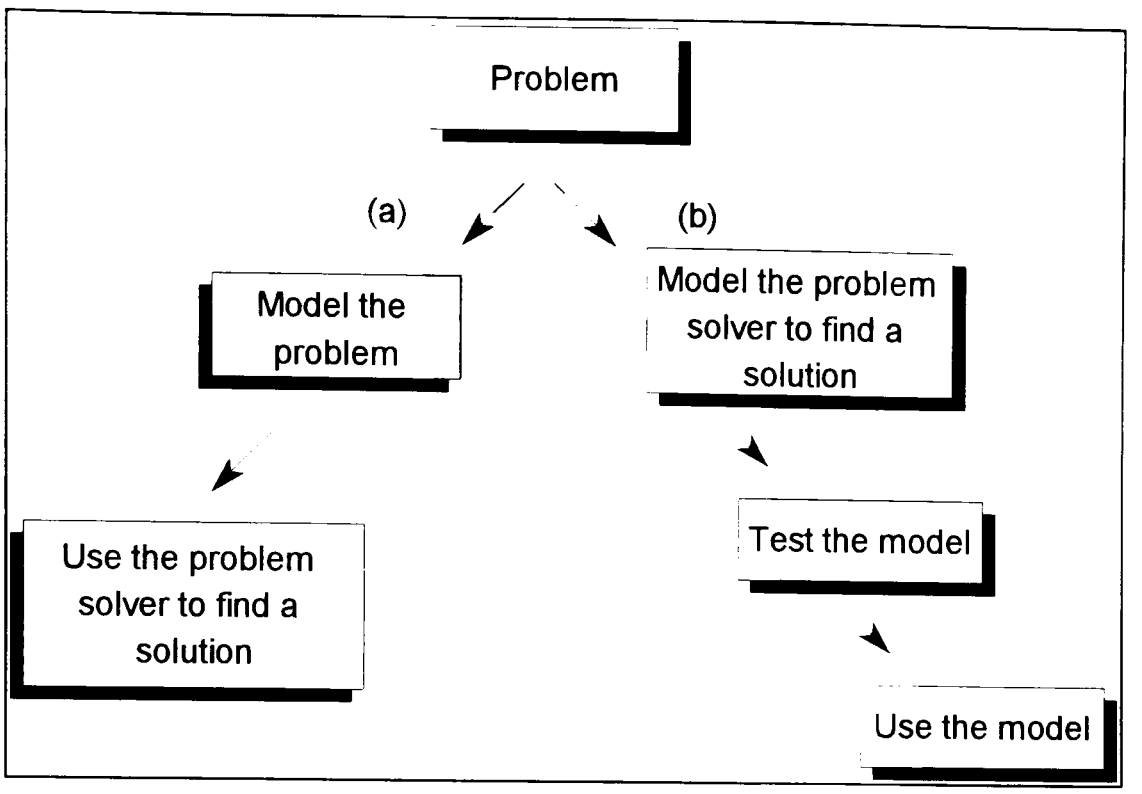


Figure 3.1: A Block diagram showing (a) the steps followed by the normal system analysis versus (b) Neural Network approach as a problem solver

3.4 Artificial Neural Networks and Control Systems

Learning algorithms are required to operate in ill-defined and time-varying environments with a minimum amount of human intervention. These techniques are typically used to control plants for which a conventional mathematical analysis is not possible, and many different learning systems have been proposed for use within IC systems. Recent research interest has re-focused on using biologically inspired learning algorithms and control architectures, and this have been part of a wider improvement or research activity into ANN. ANNs and the control community have a long history, which probably began with Wiener's Book *Cybernetics* published in the late forties [21]. During the fifties and sixties, the adaptive control field grew a notable success. The unification of several parameter estimations algorithms, coupled with the development of gradient and stability-based learning rules, provided a firm theoretical background for many of the practical

applications [22]. Several neural controllers were developed in the sixties, most notably Widrow's inverted pendulum controller in 1963 [23]. More recently, the developments of neural network architecture and learning algorithms have provided the stimulus for control engineers to re-evaluate the potential of ANN-based controllers, or *neurocontrollers*. The new ANNs were proposed by researchers from different disciplines, such as computer science, psychology, etc., and many of the learning algorithms have their 'parallels' in the adaptive identification and control fields.

The most important characteristic of ANN is its ability to learn the frequently complex dynamic behavior of a physical system. Learning is the process where the network approximates the function mapping from system inputs to outputs, given a set of observations of its inputs and corresponding outputs. This is done by adjusting the network internal parameters, to minimize the squared error between the networks outputs and the desired output. One such method is the error Back-Propagation (BP) algorithm by Rumelhart, Hinton and Williams [6], which is essentially a first order gradient decent method. The ability to approximate unknown functions through presentation of their instances makes ANN a useful tool for modelling in engineering applications. In addition, Hornik *el. al* [24] have proven that multi-layer back propagation networks have the capability of approximating any nonlinear continuous function. ANN are of interest to the control community because they have the potential to treat many problems that cannot be handled by traditional analytic approaches. Different ANN architectures are employed in which they have the capability to "learn" system characteristics, through non-linear mapping ANNs can be used to implement highly non-linear models/controllers with weights or internal parameter that can be determined by a self learning process. For control engineering, ANNs are attractive because they have the ability of non linear plant modelling, can handle large amount of sensory information, perform collective processing

and learning and offer the potential for highly parallel computation. They offer the promise of better solutions, to control problems that are so complex that analytical design techniques do not exist and may not exist for some time to come.

There have been some researches on using ANNs for the control of dynamic systems. Psaltis, Sideris and Yamamura discussed a number of interesting techniques for using the back propagation network to control plants [25]. Several techniques for process identification and control using neural network are discussed in Tariq Samad *el. al* [26]. Nguyen and Widrow have reported an ANN application to self learning of backing a trailer truck into a loading duck [27]. An ANN feed forward model was used in Jordan and Jacobs for system learning and control [28].

One particularly popular region of inquiry has been applying the neural computational paradigm to existing problems in nonparametric system identification [29], measurement prediction [30], fault detection [31] and situations where input-output data are available but functional relationships are poorly understood such as the treatment of nonlinear dynamic system with approaches to the design neural network based controllers [32], [33].

ANNs have made a significant impact on the chemical industry, with application in nonlinear process and human operator modelling, automatic plant knowledge elicitation, fault detection and monitoring, process control and optimization and sensor validation, interpretation and fusion. For example, ANNs have been used as part of model predictive control [34], for a Continuous Stirred Tank Reactor (CSTR) for pH control of sodium hydroxide, by minimizing a quadratic cost function over a finite time horizon subject to constraints on the pH range. An ANN is adapted to form a dynamic model of the CSTR

off-line, then in real-time with ANN fixed (with variable bias), an optimizer plans a series of actions/control over the planning horizon. The optimization problem is one of nonlinear programming (via feasible sequential quadratic programming) that may be readily implemented on a special chip.

An ideal application of ANNs is in the field of nonlinear system identification [35]. Virtually any discrete time nonlinear system may be represented by Nonlinear Auto Regressive Moving Average with eXogenous inputs (NARAMAX) model [33], which represents a system in terms of its delayed inputs and outputs. In general the NARAMAX model has the following form:

$$y^*(k+1) = f^*(y^*(k),\ldots,y^*(k-d_y),u^*(k),\ldots,u^*(k-d_u)) \qquad (3\text{-}1)$$

where d_u and d_y are the maximum delays in the input and output vectors u^*, y^*, respectively. This form of model is ideal for system identification purposes because the model is expressed entirely in terms of known quantities.

An ANN may be used to identify the NARAMAX model of a system by making the arguments of (9) the inputs to the ANN and making the output of the ANN be the one-step-ahead (predicted) output vector $y^*(k+1)$ as shown in fig. 3.2 (TDL denotes a tapped delay line whose outputs are delayed values of its inputs). The ANN may then be trained to emulate the function $f^*(.)$ in (3-1) by comparing the *predicted* output vector with the *actual* output vector at time k + 1, and using the *error* to update the ANN weights via the error back propagation algorithm.

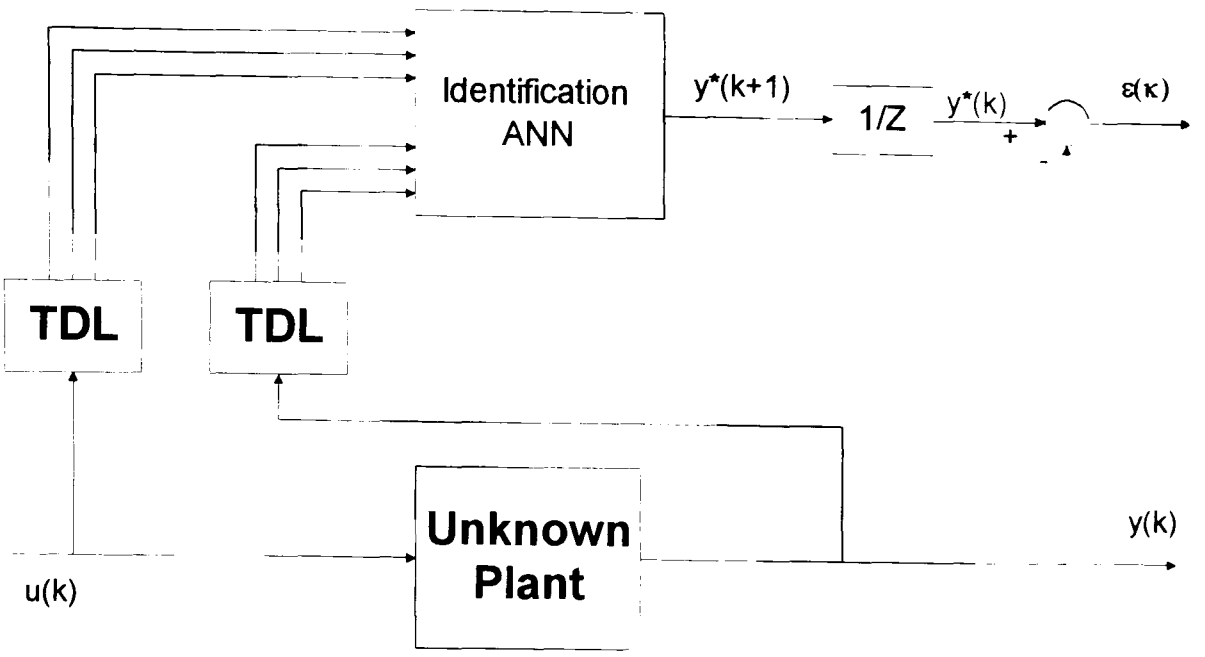


Figure 3.2: System identification using ANNs

Fundamental to the majority of intelligent control schemes is plant identification or modelling. ANNs (just like conventional nonlinear time series methods such as NARMAX, extended Kalman filters, etc.) have been applied to dynamic process identification. In the chemical process industry, ANNs have been used in the identification of waste treatment plants, with multiple holding or settling tanks [37]. The plant is typically 5 input, 9 output multivariable process. Excellent modelling results are obtained and the plant has been successfully controlled using a model predictive algorithm.

3.4.1 Neural Computing Benefits

Neural computing is *different* from conventional algorithmic computing, although the former can generally be decomposed into an algorithm and implemented on a serial machine. These apparently contradictory statement can be resolved if it is accepted that it is the *approach* which distinguishes the two techniques, rather than the final implementation. Neural networks offers solutions to problems that are very difficult to

solve using traditional algorithmic decomposition techniques and the potential benefit of a neural approach are :

- learning from the interaction with the environment (learn by experience), rather than by explicit programming (modelling);
- few restrictions are placed on the type of functional relationship that can be learnt;
- ability to generalize (interpolate and extrapolate) the training information to similar situations; and
- inherently parallel and the computational load can be evenly distributed across many simple processing elements. Thus the networks posses some degree of fault tolerance with respect to processor failures.

The first three properties are desirable for any learning algorithm, and the fourth can be used to apply these networks to larger real-time systems. If a learning algorithm possesses these properties, it can endow the control system with the following advantages [38]:

- decreasing the required amount of human intervention;
- increasing the flexibility of the control system;
- improving the performance of the control system; and
- reducing the initial design time and cost.

The performance of an ANN (learning, recall, computational burden, etc.) depends on how well satisfies the first property list, and this determines its potential for off-line design problems. For on-line adaptive modelling and control the algorithm also need to posses the following properties:

- learn significant new information (plasticity) in a stable manner and in real-time while retaining knowledge previously learned; and
- provable learning convergence conditions for local & global optimization.

Some of the most commonly used ANNs satisfy neither of these properties, although this means that while they can be used in neurocontrol application, learning should only occur off-line. In chapter 4 a technique based on GENE approach is developed which is suitable for on-line global error evaluation.

Chapter 4 Artificial Neural Networks

4.1 Introduction

The primary objective of the control system of MSF is to determine the capability of the process to remain in the stable mode of operation particularly during load change (transient conditions) as well as when disturbances such as shortage of steam supply is imposed. Control system design for MSF desalination plants requires a knowledge of the inherent plant dynamic behavior that have quite a few complication process characteristics such as time delays, disturbances, unmeasured variables, time-varying parameters, nonlinearities and multivariable interactions. One of the really crucial flow- related control parameters are not those which are directly observable (such as temperature, flow, and pressure). Instead, they are the more indirect quantities such as overall (and localized) heat transfer coefficients in large tube bundles, non-equilibrium losses in flashing of brine, and the stability of tray brine levels and inter-stage flows. Measurement of these phenomena must be derived from the direct measurements often requiring considerable accuracy. While microcomputer based controllers are adequate hardware that is adequate theoretically to overcome these problems, the challenge is to find the appropriate software to direct the hardware.

MSF desalination plants are characterized as a complex multi-variable process with several regulated variables and interacting loops. The control loops are usually controlled separately by PI and PID controllers and the problem of interaction exist such that the adjustment of a single set-point causes a profound influence on many other control loops in the process. By multi-variable, we refer to those processes wherein many, strongly interacting variables are involved. The control input (set point) can be calculated from a steady state mathematical model, which is based on the basic equations describing the behavior of the desalination unit (heat and mass balance equations and the heat exchange equation of each stage). This model is valid only for the steady or near steady state situation. The control algorithm is required to be capable to manage all situations taking place during load transients, and therefore requires a dynamic nonlinear model representation. It is often very difficult and time consuming to drive a realistic system model, especially when the basic mechanism of the process is not completely understood. Computationally expensive calculations are required for the desired precision and accuracy. This aspect is specially crucial for enhanced real-time performance. As a result, control algorithms are often synthesized based upon their linear approximation. However, MSF systems are nonlinear, consequently, un-modeled dynamics and robustness problems arise. Therefore in practical applications supervisory control is adopted [22].

Apart from the above methods, fast assessment of the control system of MSF based on Artificial neural networks (ANNs) can be an alternative method. In this alternative approach, the capabilities of the ANN to learn and generalize enables the network to obtain complex mapping of the MSF dynamic behavior and the various transient control actions. ANN can estimate the dynamic behavior and the control actions required due to a disturbance in negligible time. Recently, ANN approaches have been proposed for the estimation of the control actions of MSF plants [39], [40] &[41].

In the recent years, neural network based control system has been receiving more and more attention because they can handle large amount of sensory information, perform collective processing and their ability to capture the approximate nature of the real system with particular attention to the dynamic nonlinear processes. The ANN can be traced back to the 1940's when papers by McCulloch and Pitts on the modelling of the neuron [42] and Hebb on learning appeared [43]. Rosenblatt's publication on the Perceptron (1958) and Widrow's ADLINE (1960) sparked a great deal of interest initially [44], [45]. However, there was no known learning algorithm that could be applied to adjust the weights for a specific calculation. In 1969 the text, Perceptrons, was published by Minisky and Papert and it showed that the linear neural networks that had received so much attention were severely limited in the problems they could solve [46]. This text caused most people to lose interest in neural computation except few dedicated scientists. Gradually, a theoretical foundation emerged, upon which the more powerful *multi-layer networks* of today are being constructed with many impressive demonstrations of ANN capabilities where the most have used the famous algorithm Back Propagation (BP) to train the networks, perhaps the most successful of the current algorithm. Back Propagation invented independently in three separate research events (Werbos 1974, Parker 1982; and Rumelhart, Hinton and Willliams 1986) [**47**] [48] [6], provides a systematic mean for training multi-layer networks, thereby overcoming limitation presented by Minisky. Werbos developed the back-propagation algorithm first in 1974, but his achievement remained almost unknown [47]. Parker rediscovered the technique in 1982 [48], and independently in 1986 by Rumelhart, Hinton and Williams [6]. The multi-layer Perceptron networks were not used in the past because of lack of an effective learning algorithm. This has recently changed, mainly owing to Rumelhart and his co-workers who have popularized the back-propagation algorithm among the scientific community [6, 49].

Various types of neural network are being studied or used in applications. However the most popular neural network is the feed forward neural network with BP learning algorithm. Applications of feed forward neural networks have been the task of "learning" maps from discrete data. Examples of such map learning problems can be found in areas such as modelling chemical process systems [29], control and identification of dynamically systems [32], control of nonlinear systems [25], to name a few. In most of these applications, feed forward neural networks act on data by detecting some kind of underlying organization. They learn (closely approximate) the similarities among patterns directly from instance of them to give the desired map. That is they infer solutions from data without prior knowledge of the regularities in the data. This is useful because gathering data does not require explaining it. They extract the regularities from data empirically. Thus, they can bridge the gap between an individual example and general relationships. This is a valuable characteristic for appreciation to predict the nonlinearinties and uncertainties for the desalination process, where no obvious expression for calculating them.

Artificial neural networks (ANN) typically consist of an architecture with many simple computational elements or nodes arranged in layers and operating in parallel. The weights, which define the strength of connection between nodes, are adapted during the learning to yield good performance. The term *neural* reflects the fact that the initial inspiration for such networks was derived from the observed structure of biological neural processing systems. Changing the weight of an element will alter the behavior of the element and, therefore, will also alter the behavior of the whole network. The arrangement of the network's nodes and connections defines its architecture and there are many possible variations.

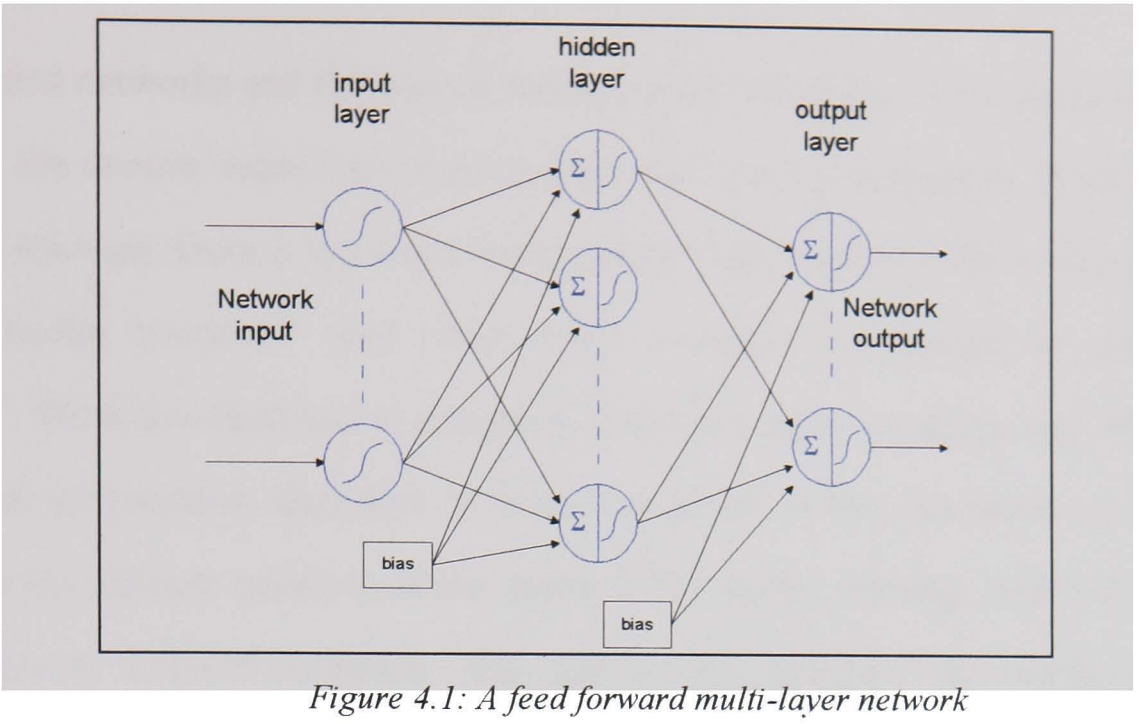


Figure 4.1: A feed forward multi-layer network

where each circle corresponds to a node and each arrow represents a weighted link

One popular arrangement is shown in figure 4.1 where the nodes are arranged into *layers* and each node in one layer has connections only to with nodes in the preceding layers. The goal here is to choose the weights of the network to achieve a desired input/output relationship. This process is known as training the network. In other word the weights are adapted by training to improve *performance*. The network can be considered memoryless in the sense that, if one keeps the weights constant, the output vector depends on the current input vector and is independent of past inputs [6]. This kind of network is known to be capable of learning complex input - output mappings. That is, given a set of inputs and desired outputs or targets, an adequately chosen neural network can emulate the mechanism in order to reproduce the data set through learning.

The success of any ANN approach for MSF dynamic control depends on the successful learning of the correct mapping. ANN models are specified by three elements: neuron characteristics, network topology and training rule. There are two classes of ANN structures, namely feed forward and recurrent. In the present work, Multi-layered feed forward networks (MFN's) with error back propagation learning algorithm is adopted.

Feed forward networks and the neuron characteristics are discussed in section 2. Section 3 reviews the famous supervised training algorithm back propagation. When an ANN is applied to solve problems it is always worth asking "what size of a network should we use for a particular problem?" And "how many examples are needed to generalize the problem?". These are discussed in section 4. There are some variations and improvements to the back propagation algorithm to avoid the pitfall of the saturation of the network nodes and the ultimate paralysis of the entire MFN during learning; and the problems of convergence to a *local minimum*. The last section develops the methodologies that account for generalization and avoiding local minima so that a consistent ANN approach can be developed to reproduce the MSF process behavior. The methodology when used could predict the brine levels in both the first and last stages. The methodologies are developed based on *global error node evaluation* scheme for MFN. The development of a dynamic model and with an example for set point generation that can reproduce the process behavior using the standard back propagation as well as using the GENE approach for the MSF desalination plant are described in chapters 5 & 6.

4.2 Feed Forward Networks

4.2.1 The Neuron and the Activation Function

The basic component in a feed forward network is the single "neuron" model depicted in figure 4.2 (a), where $x_1,....,x_n$ are the inputs to the neuron, $w_1,....,w_n$ are the multiplicative weights applied to the inputs, I is a biasing input, g : R ➔ R, and S is the output of the neuron. Thus

$$S = g(\sum_{i=1}^{n} w_i x_i + I) \qquad (4\text{-}1)$$

or $$S = g(\Sigma) \qquad (4\text{-}2)$$

The "neuron" of figure 4.2 (a) is often depicted as shown in figure 4.2 (b) where the input weights, bias, summation, and function g are implicit. The activation function may be:

(a) simple linear function, $g = C(\Sigma)$, where C is a constant,

(b) a threshold function, $\begin{matrix} S = 1 \text{ if } \Sigma > T \\ S = 0 \text{ otherwise} \end{matrix}$, where T is a constant threshold value,

or a function that more accurately simulates the nonlinear transfer characteristic of the biological neuron and permits more general network functions.

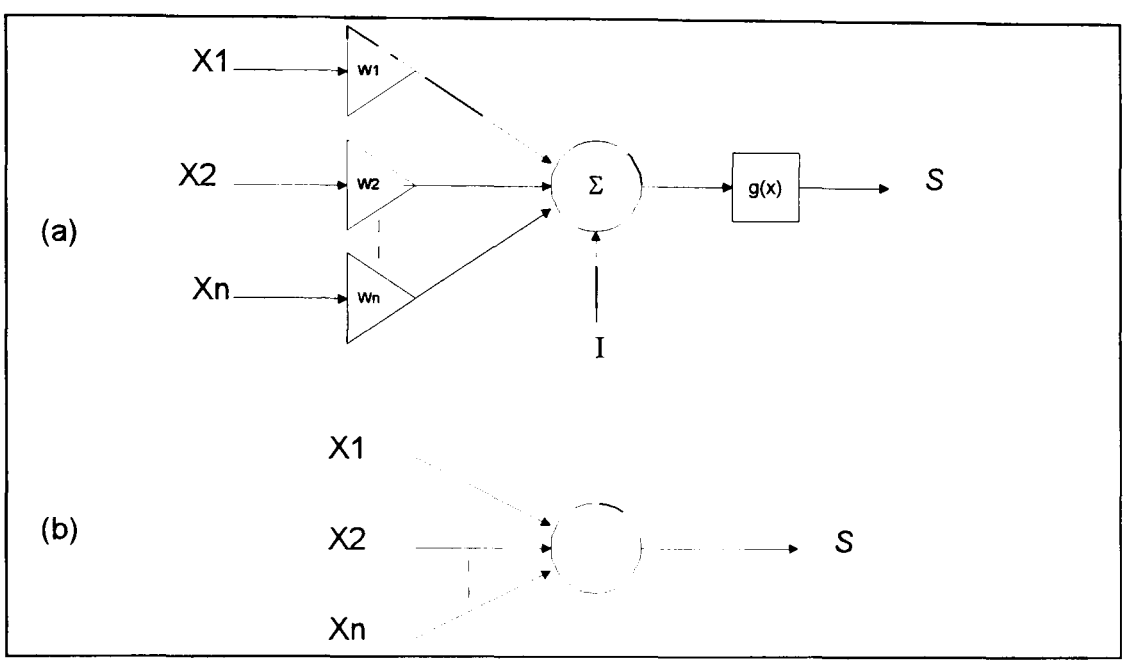


Figure 4.2: (a) Single neuron model, (b) Simplified schematic of single neuron

4.2.1.1 The Squashing Function

If the $g(\Sigma)$ function compresses the range of Σ, so that S never exceeds some low limit regardless of the value of Σ, g is called a *squashing function*. The squashing function is often chosen to be the logistic function or "sigmoid" (meaning S-shaped) as shown in figure 4.3 and expressed mathematically as

$$g(x) = \frac{1}{1+e^{-x}} \qquad \text{thus} \qquad S = \frac{1}{1+e^{-\Sigma}} \qquad (4\text{-}3)$$

Another commonly used activation function is the hyperbolic tangent. It is similar in shape to the logistic function but symmetrical about the origin, resulting in S having the value 0 when Σ is zero (see figure 4.3), and is expressed as follows:

$$S = Tanh(\Sigma) = \frac{1+e^{+\Sigma}}{1+e^{-\Sigma}} \qquad (4\text{-}4)$$

The squashing activation function can be thought defining a nonlinear gain for the artificial neuron. This gain is calculated by finding the ratio of the change in S to a small change in Σ. Thus, gain is the slope of the curve at specific excitation level. It varies from low value at large negative excitations (the curve is nearly horizontal), to a high value at zero excitation, and it drops back as excitation becomes very large positive. Grossenberg [50] found that this nonlinear gain characteristic solves the noise-saturation dilemma that he posed; that is how can the same network handle both small and large signals. Small input signals require high gain through the network if they are to produce usable output; however, a large number of cascaded high-gain stages can saturate the output with the amplified noise (random variation) that is present in any realizable network. Also large input signals will saturate high-gain stages, again eliminating any usable output. The central high gain region of the logistic function solves the problem of processing small signals, while its regions of decreasing gain at positive and negative extremes are appropriate for large excitations. In this way, a neuron performs with appropriate gain over a wide range of input levels.

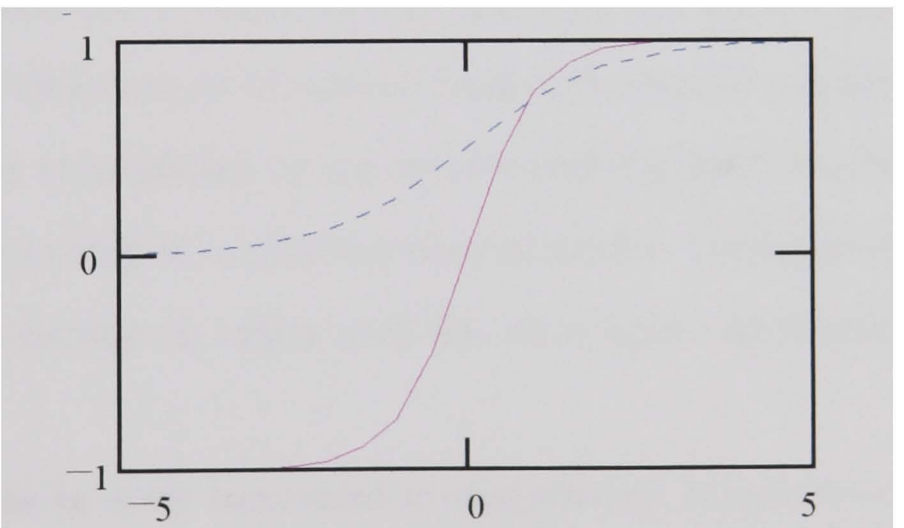


Figure 4.3: The neuron activation function using (a) the tanh function, (b) the sigmoid function

4.2.2 Multi-Layer Feed Forward Network (MFN)

A feed forward network is constructed by interconnecting a number of neurons (such as shown in figure 4.1) so as to form a network in which all connections are made in the forward direction (from input to output without feed back loops). Neural networks of this form are usually composed of an input layer, a number of hidden layers, and an output layer. The input layer consists of neurons that accept external inputs to the network. Inputs and outputs of the hidden layers are internal to the network, and hence the term "hidden". Outputs of the neurons in the output layer are the external outputs of the network. Once the structure of the feed forward network has been decided, i.e., the number of hidden layers and the number of the nodes in each hidden layer have been set, a mapping is learned by varying the connection weights w_{ij} and the biases, I_j so as to obtain the desired input-output response for the network.

4.3 Training Algorithm

Finding suitable parameters for ANNs by an experiment has been an impediment to the development of ANNs, especially when some of the algorithms consume large amount of computer time. A network is trained so that application of a set of inputs produces the desired (or at least consistent) set of outputs. Each such input (or output) set is referred as a vector. Training is accomplished by sequentially applying input vectors, while adjusting network weights according to a predetermined procedure. During training, the network weights gradually converge to values such that each input vector produces the desired output vector.

Training an ANN can be either *supervised* or *unsupervised*. In unsupervised training there is no feedback from the environment to determine what that output should be. The network must discover for itself the patterns, features, regularities, correlations, or categories in the input data and code them in the output. The neurons and connections must thus display some degree of self-organization. Kohonen's feature-map forming nets

[51], the classical K-means [52] and leader clustering algorithms [53] are trained without any supervision. On the other hand, supervised training is to make the network output equal to the desired target function for any input. The standard approach is to

(1) Define an error function measuring the difference between the target and actual output function,

(2) Determine how changes in the network weights (parameters) affect the error.

(3) Adjust the weights in a way to that reduces the error.

4.3.1 Back-Propagation Learning Algorithms

The Back propagation (BP) gives a prescription for changing the weights in any feed-forward network to learn a training set. The basis of the algorithm is the gradient descent. It can be considered as an unconstrained optimization problem of a suitably constructed error function (cost function). Typically, the error function is the sum of squared differences between the desired target $y(x_k)$ and the actual network outputs $S(x_k)$ are summed over all training pattern k,

$$E(w) = \sum_k (y(x_k) - S(x_k))^2 \qquad (4\text{-}5)$$

The core of the algorithm is a repeated loop in which

(1) The derivative chain rule is applied to determine how weight changes affect the error

(2) The weights are adjusted by small increments in the direction that reduce the error.

In "*batch mode*", every training pattern is considered before each weight change, and the algorithm approximate gradient descent when the step size is small enough. In "*on-line mode*", a random subset of patterns (usually just one) are considered before each weight change. When the step size is small enough, this approximates gradient descent since the accumulated weight changes tend to average to the true (negative) gradient. During training, an equation is used to minimize the sum of the network's squared errors. The

minimization process has an intuitive geometric meaning. All possible sets of weights can be plotted against the errors. The result is an error surface shaped like a bowl, whose bottom marks the set of weights with the smallest error. An idealized two-dimensional error surface is shown in Figure (4.4). However, real error surfaces typically have ravine-like features and dent-like local minima. Finding the bottom is the goal during training [54].

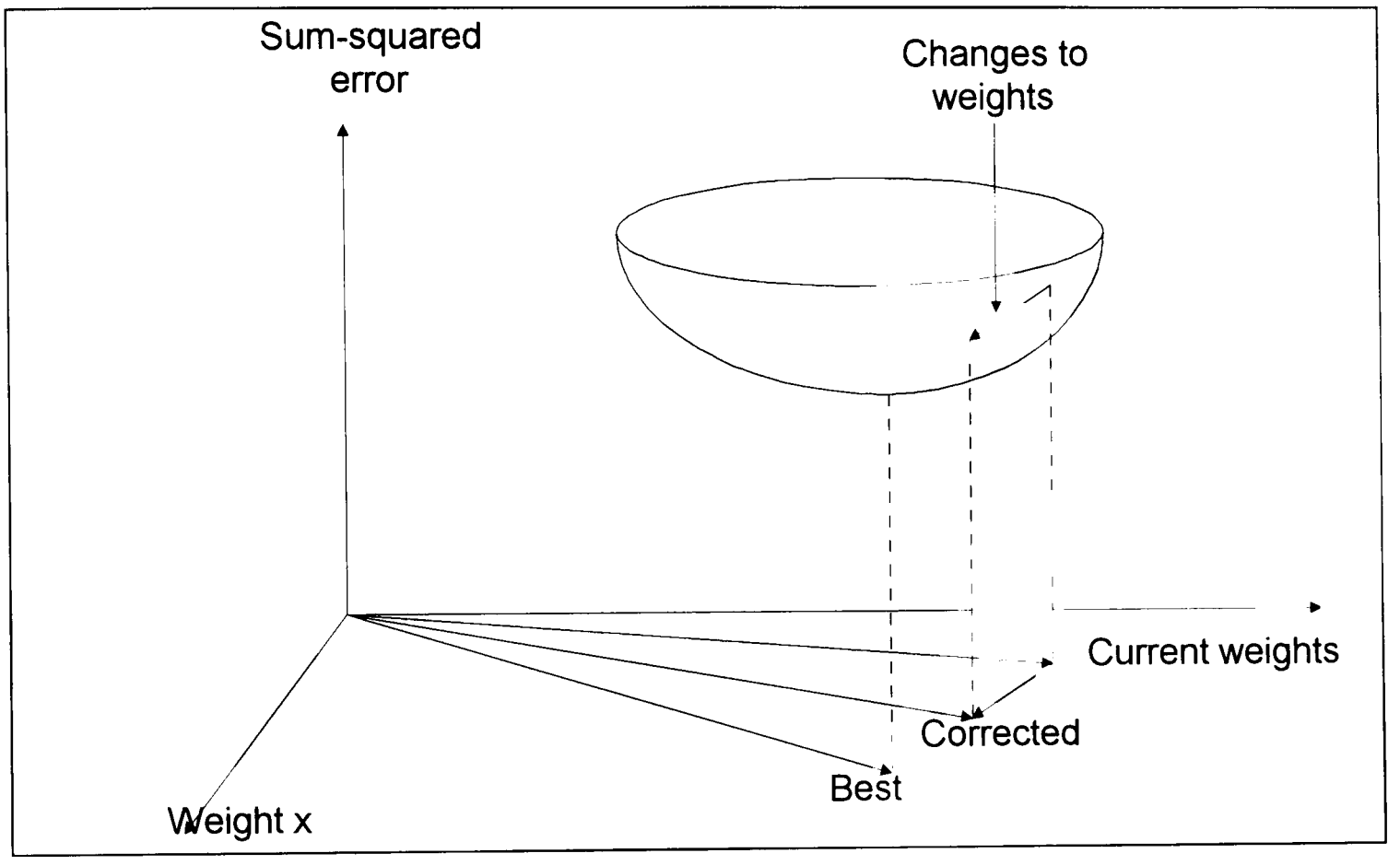


Figure 4.4: An idealized two dimensional error surface

There are two requirements for the BP algorithm:

- The connections of the ANN must be in the forward direction, i.e., from a neuron closer to the input layer to a neuron in another layer closer to the output layer.
- The nonlinear function of each neuron must be continuous and differentiable. The tanh function as shown in figure 4.3 (a) is used here.

The back propagation can be summarized as follows:

step 1. *Initialize weights and thresholds*

Set all weights and thresholds to small random numbers

step 2. *Present inputs and desired outputs*

Present inputs $x_0, x_1, \ldots, x_{n-1}$ and specify the desired outputs $y_0, y_1, \ldots, y_{m-1}$. The net input to the j^{th} neuron of layer l, $1 \le l < L$, at time k is given by

$$x_j^l(k) = \sum_{i=0}^{N_{l-1}} w_{ij}^{l-1}(k) s_i^{l-1}(k) \qquad (4\text{-}6)$$

In the above equation, it is assumed that $s^l_0 = 1$ for all l thus w^l_{0j} is the bias for the j^{th} network neuron in layer $l+1$. The output of a network neuron will be

$$s_j^l(k) = g(x_j^l(k)), \quad 1 \le l \le L \qquad (4\text{-}7)$$

For units in the output layer, the net input is given by

$$x_j^l(k) = \sum_{i=0}^{N_{L-1}} w_{ij}^{L-1}(k) s_i^{L-1}(k) \qquad (4\text{-}8)$$

step 3. *Calculate actual outputs*

Use Equations 4-7 to calculate actual outputs $S_0, S_1, \ldots, S_{m-1}$

step 4. *Adapt weights*

Use recursive algorithm starting from output neurons and working back to the hidden neurons. Weights are adjusted according to:

$$w_{ij}^l(k+1) = w_{ij}^l(k) + \eta d_{jk}^l, \qquad (4\text{-}9)$$

where the scalar $\eta > 0$ determines the length of the step to be taken in the direction of the vector $d_{ik}^l \cong \sum_{j=0}^{N_{l+1}} \Delta w_{ij}^{l+1}(k)$. The BP adopts the steepest descent (gradient) method and defines the directions as

$$d_{ik}^l \equiv -\nabla E(w_{ij}^{l+1}(k)) \qquad (4\text{-}10)$$

and the weight update equation can be rewritten as

$$w_{ij}^l(k+1) = w_{ij}^l(k) - \eta e_i^{l+1}(k) S_i^l(k) \qquad (4\text{-}11)$$

or using $\Delta w_{ij}^{l}(k+1) = [w_{ij}^{l}(k+1) - w_{ij}^{l}(k)]$ (4-12)

then $\Delta w_{ij}^{l}(k+1) = -\eta . e_{i}^{l+1}(k) . S_{i}^{l}(k)$ (4-13)

then by smoothing the weight changes by over-relaxation [6], i.e. by adding the momentum term

$$\Delta w_{ij}^{l}(k+1) = -\eta . e_{i}^{l+1}(k) . S_{i}^{l}(k) + \alpha . \Delta w_{ij}^{l}(k) \quad (4\text{-}14)$$

where $0 \le \alpha < 1$ (typically $\alpha = 0.9$)

If neuron j is a hidden neuron then the error is

$$e_{j}^{l} = g'(x_{j}^{l}(k)) \sum_{p=1}^{N_{l+1}} e_{p}^{l+1}(k) w_{jp}^{l}(k), \quad 1 \le l < L \quad (4\text{-}15)$$

If neuron j is an output neuron then the error

$$e_{j}^{L} = (S_{j}^{L}(k) - y_{j}(k)) g'(x_{j}^{L}(k)) \quad (4\text{-}16)$$

The above equation represents the standard BP of error, $g'(.)$ is the derivative of the activation function of the network neuron. To employ the BP algorithm for updating connection weighs, the activation function g'(.) of the network nodes must be differentiable. In the present work, the following tanh function is adopted for the activation function of any hidden layer node i

$$Tanh(x) = \frac{1 + e^{+x}}{1 + e^{-x}} \quad (4\text{-}17)$$

Appendix A gives detailed derivations of equation (4-12) and (4-13).

step 5. *Repeat by going back to step 2*

This procedure is repeated until a convergence criterion is met or a specified number of iterations are reached. The convergence criterion can be either that the error is less than specified value or that the weights have stabilized.

The weight updates described above are used for *on-line learning* where a learning example is presented at the input and then all the weights are updated before the next learning example is presented. This description is based on the Generalized Delta Rule. The error in the output layer is computed as the difference between the desired output (y) and the actual output (S). This error, is transformed by the derivative of the transfer function, is "back-propagated" to prior layers where it is accumulated. This back-propagated and transformed error becomes the error term for that prior layer.

Alternatively in the *batch learning* (using the cumulative delta rule) the weight changes are accumulated over some number of the learning examples before the weights are actually changed. The number of training presentations between weight updates is length of an epoch. One of the problems with the cumulative delta rule is the linkage between the learning rate and the epoch size. Adjusting the epoch requires adjusting the learning rate. One way to overcome this is by using the normalized cumulative delta rule; dividing the learning rate by the square root of the epoch size.

In spite of algorithm apparent simplicity, the algorithm has proven remarkably effective, and there are many examples of network trained to implement relatively complex functions. This is not to say that difficulties never occur. BP training is often very slow to converge in training a MFN and may converge to sub-optimal solutions. The convergence proof of the BP is based on calculus limited theory and therefore requires infinitesimal weight adjustments. BP is based on the method of steepest descent to update weights. At each iteration of the steepest descent procedure, the weight values are modified in the

direction in which the error function decreases most rapidly. This direction is given by the derivative of the error surface at the current point in weight space. The magnitude of adjustment in a weight is proportional to the partial derivative of the error function with respect to that weight. The error surface may posses certain nonlinear properties which can cause convergence difficulties if the steepest descent algorithm is used. From this viewpoint, problems related to the method can be categorized into two groups. Firstly, problems associated with the various initialization parameters such as architecture, number of layers, number of nodes, learning rate selection and weight initializations scheme, etc. The second group of problems is related to the final results; such as local minima solution, overfitting and network paralysis that can result in outright failure of the network. Practical techniques for solving the first group of problems are discussed in the following section. Some remarks are very basic and may be viewed as a checklist of standard procedures. Others are more specific. Many of the remarks apply to any learning system, but unless otherwise stated, the focus is on supervised learning in MFN. To address the second group of problems, a method based on the *global error node evaluation* (GENE) is developed for MFN to improve learning ability of BP in a subsequent section in this chapter.

4.4 Problems with the BP and their Enhancement

4.4.1 Data Preparation and Preprocessing

Neural networks are often trained from examples of a desired input - output relationship. It is important that these examples adequately describe the function. For this, the data is required to be distributed to cover the information about the relative importance of different regions of the function, it should generally match the distribution of function behavior that will occur in normal operation. The next step usually required is re-scaling of

the data such that the variables with different ranges will have an equal footing. This can be by centering and normalization of the variables. A commonly used normalization

$$x' = (x-\mu)/\sigma \qquad (4\text{-}18)$$

where μ is the mean value of x and σ is the standard deviation [71]. Normalization is based on minimum and maximum values. Such preprocessing is an essential factor to overcome the poor convergence rates of MFNs.

4.4.2 Architecture Selection

The numbers of input and output layer nodes are usually problem dependent. The problem lies in the selection of the hidden layer node number, which is directly related to the number of weights to be adjusted during training. The learning speed and generalization characteristics of ANNs are dependent on their architectures. The goal is to find a network powerful enough to solve the problem, yet simple enough to train easily and generalize well. Thus, the viability of a specific architecture can only be evaluated after training. Two approaches are possible. The first approach is to train many different architectures on the same problem and to use the one with the best post-training characteristics. This approach significantly increases training time since many ANNs must be trained. Furthermore, an optimal architecture is not necessarily obtained with this technique since it may not be one of the initial selections. The second approach is that the optimization of the network architecture becomes part of the training objective. This can be either by *Pruning* or *Constructive* techniques. The pruning approach is to train a network that is larger than necessary and then remove unnecessary parts. The large initial size allows reasonably quick learning with less sensitivity to parameters, while the reduced complexity of the trimmed system favors improved generalization. In several studies, pruning techniques have produced small networks that generalize well where it was very difficult to obtain a

solution by training a small network (obtained by pruning) from scratch with random weights [55].

Many pruning techniques have been suggested; a survey can be found in Reed (1993) [56]. Many of the algorithms fall into two board groups. One group estimates the sensitivity of the error to removal of elements and removes these with the least effect. Another group adds terms to the error function that penalize unnecessary complex solutions. In general, sensitivity methods modify a trained network; the network is trained, sensitivities are estimated, and then elements are removed. Penalty methods modify the cost function so the optimization drives unnecessary weights to zero and, in effect, removes them during training. Even if the weights are not actually removed, the network acts like a smaller system. An advantage is that training and pruning are done in parallel so the network can adapt to minimize errors introduced by pruning.

Although pruning and penalty term methods often may be faster than searching for and training a minimum-size networks, they do not necessarily reduce training times; larger networks may take longer to train because of sheer size, and pruning takes some time itself. The goal, however, is improved generalization rather than faster training speed.

The opposite approach to pruning is to build the network incrementally by adding elements until a suitable configuration is found. Starting with a small network, train until the error stops decreasing and then add a new node (or nodes) and resume training, repeating until an acceptable error is achieved. In some cases, constructive methods can be faster than pruning methods, since significant learning may occur while the network is still small. The approaches are not incompatible and often used together. Since constructive method, when used alone, sometimes create larger networks than necessary, a follow-up pruning phase can be useful to reduce the size.

It should be noted that pruning and constructive techniques are a means of adjusting network size rather than a way of deciding what size is appropriate. Other criteria are often useful to decide when to stop adding or removing elements. Bounds on the number of hidden neurons were studied by Huang [57] and Sartori [58]. It was proved that k-1 (k is the number of inputs) is the least upper bound on the number of hidden neurons needed to realize an arbitrary real-valued function defined on a finite and linearly separable set.

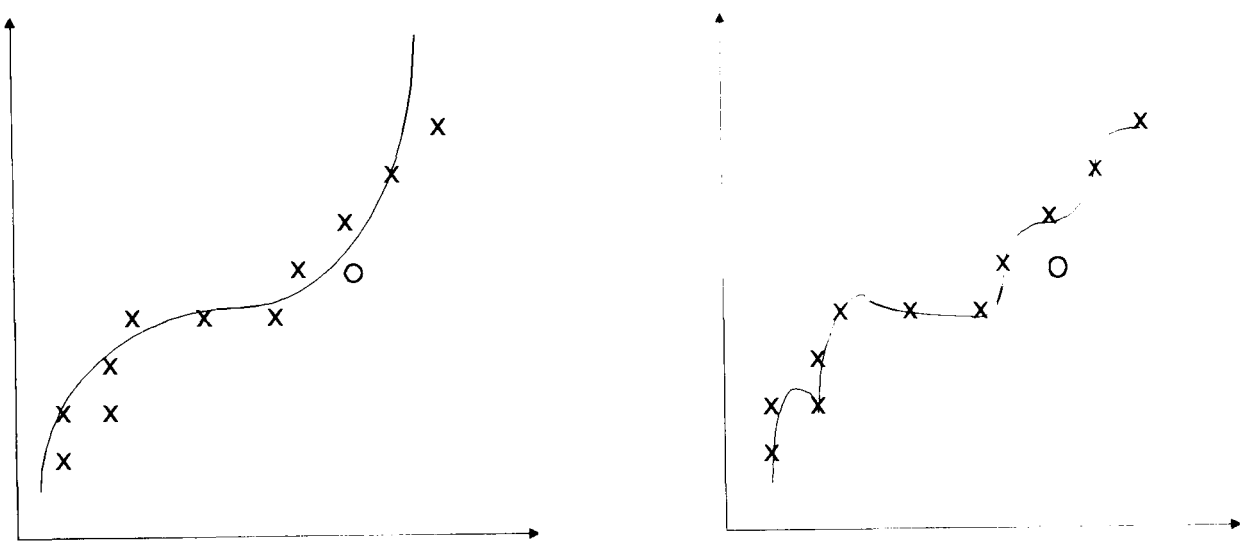


Figure 4.5: (a) A good fit to noisy data indicated as crosses. (b) Overfitting of the same data: the fit is perfect on the training set, but could be poor on a test set represented by the circle

It is generally possible to get increasingly better performance on the training set by increasing the complexity of the model, i.e., increasing the number of hidden neurons and thus the number of weights, but such a procedure does not necessarily lead to a better ability to generalize. Too many free parameters could results in *overfitting*. As illustrated in figure 4.5 a curve fitted with too many parameters follow all the small details but is very poor for interpolation and extrapolation. The same thing happens to a multi-layered network: too many weights give poor generalization [59].

4.4.3 Weight Initialization

The values of the initial weights are very important. If they are too large the squashing activation function will saturate from the beginning, luring the network to *Paralysis*. Paralysis occurs when nodes are driven into saturation. A reasonable strategy for weight initialization is to set weights to "small" random values. The randomness is intended to break the symmetry, while "small" weights are chosen to avoid immediate saturation [6]. The weight are chosen within a range such that the magnitude of the typical inputs to neuron i is less than, but not too much less than unity. This can be achieved by taking the weights w_{ij} to be of the order of $1/\sqrt{k_j}$ where k_j is the number of i's which feed forward to j (the fan-in of neuron j) [59].

4.4.4 Shortening Learning Times

There are at least two reasons for slow convergence. One is because of the variable magnitude of the components of the gradient vector. The magnitude of weight update is proportional to the derivative of the error surface with respect to that weight. If the error surface is fairly flat along the weight dimension, the derivative of the weight is small in magnitude. Thus the weight is adjusted by a small amount and many steps are required to achieve a significant reduction in errors. Alternatively, if the error surface is highly curved along the weight dimension, the derivative of the weight is large in magnitude. Thus, the value of weight is adjusted by a large amount and the value may overshoot the minimum of the error surface. The other cause for slow convergence is that the direction of the negative gradient vector may not point directly towards the global minimum of the error surface [60].

Adaptive learning rates

In equation (4-11) the symbol η is the learning rate and it is usually a real number between 0.1 and 1. To increase the rate of convergence and stability of the learning process, a

momentum term [6] is usually added to the right hand side of equation (4-11). This term is given by the product of the increment of weight change $\Delta\omega_{ij}$ of the preceding pattern presentation and a constant α as in equation (4-14).

4.5 Analysis for BP Algorithm and *GENE* Approach

4.5.1 Introduction

Most multi-layered feed forward networks (MFN) that adopt the sigmoidal or hyperbolic tangent function as the activation function are trained by the standard BP learning algorithm or a variant of BP [59], [6], [61]. The required *function approximation* is achieved through the learning process where the aim is to minimize the network error E of the network by modifying the weights. Such a goal is normally based on the knowledge of local error at each hidden neuron. One of the reasons for the increasing number of local minima is that the hidden layer is trained to the local error associated with the node and not the network error [62], [63], [64]. However, researchers have tried to understand and improve two characteristics of the method: (1) generalization, the ability of a network to predict the output (Y) accurately outside the original training set; (2) *learning speed* or *convergence rate*, which is vital for systems learning from real-time experience (rather than fixed training set). The function approximation could be achieved if sufficient data is available for training. But practically, it needs the network to be trained with part of the available data, and could generalize the solution. Also when the network is trained with further new data, it should not forget the previously learnt data but learn the new one as well. If this could be achieved, then the function learned can be considered as an approximation and not just an interpolation of the data. For this, a method based on *Global Error Node Evaluation (GENE)* approach for Multi-layered Feed forward Network (MFN) is developed. It retains the function approximation requirements for the

back propagation (BP) learning algorithm for a nonlinear dynamic behavior. This approach appears to be effective for the input - output modelling of complex process systems and therefore on-line adaptation is possible (when the characteristic of the system is changing or when more training data are available for another operating range).

The development of GENE approach is by addressing two problems in BP, namely, the saturation of the network nodes and the ultimate paralysis of the entire MFN during learning; and the problems of convergence to local minima. In this approach the architecture is modified by adopting linear activation nodes at the output layer with fixed weights, while the hidden layers (two layers) are having nonlinear activation nodes. The *GENE* approach is validated using the relationship of the back propagation errors between each layer (output & hidden layers), and the subsequent weights update relation during the whole learning process. The methodology when used could predict the brine levels in both the first and last stages for MSF plant more accurately compared to the standard BP. The development of a dynamic model with an example for set point generation that can reproduce the process behavior using the standard back propagation as well as using the GENE approach for the MSF desalination plant will be described in chapters 5 and 6 respectively. The developed algorithm is used for the evaluation of the critical loops due dynamic disturbances for MSF desalination plant and is discussed in chapter 6. Simulation examples from random generated values are presented in this section.

4.5.2 Principles of Error Transformation in BP

Consider that back propagation of the error is an error *scaling* process and the *transformation* of this scaled error by multiplying it by the connecting weight during back propagation to the hidden nodes. To be more specific, the error value *back propagated* from the output layer is the error *scaled* by the derivative of the neuron transfer function in the output layer. These *scaled* errors from the outer layer are then *transformed*

(multiplied) by the respective weight connection to the hidden neuron and are then added together, producing the summation term for that hidden neuron. The summation term is then *scaled* (multiplied) by the derivative of the given neuron transfer function at its current operating point to form the local error for that neuron.

This is the same as if the back-propagated error E is calculated by a *transformation* of a set of output *scaled* error E_n through the inter-layer's connection weights into a set E_m; where n, m represent the total number of neurons in each layer respectively. The *transformation scheme* is based on the values of the connection weights between the output and hidden layers. As the connection weights are updated every learning cycle, the local error evaluation for each hidden neuron will have a different *transformation scheme* for each iteration and therefore subsequent weights update will have a different minimization criteria (minimize the local node error).

4.5.3 Node Saturation

Node saturation is not required during weight updates for all nodes. All nodes are required to contribute to the function approximation. While both tanh and sigmoid functions are monotonic, from the conditioning point of view, the tanh function is the preferred choice as it lies between [-1,1], while the sigmoid is [0,1]. However, to cover the linear part of the tanh function, data scaling is usually performed to [-0.8,0.8]. The tanh derivatives are closer to unity (in a bounded region) and this decreases the distortion of the network error back propagated between the layers throughout the summation term [65].

To learn the function approximation, the *GENE* approach is developed by using two hidden layers with nonlinear activation function (tanh). A fixed weight from the hidden layers to the output layer is used. The output layer adopts linear activation nodes and is acting as a fan-in for the error to the hidden layers during error back propagation and this

satisfies the transformation requirements for the network error when fed back to all hidden nodes.

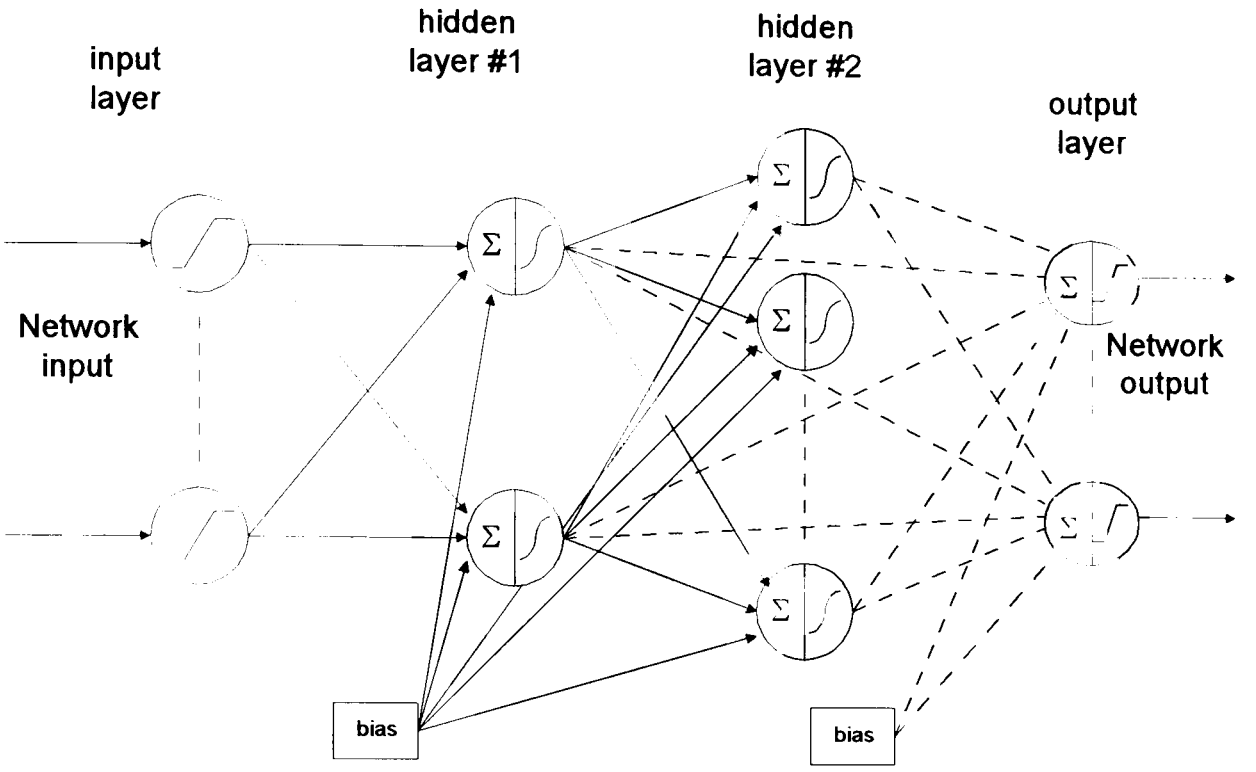


Figure 4.6: Architecture of GENE approach

4.5.4 Description of the *GENE* Approach

To overcome the local error calculation of network hidden layer nodes as well as the ultimate node saturation or paralysis of entire MFN's during training, the *GENE* approach is proposed and based on the following:-

(a) A variant to the back propagation learning algorithm to perform the same *transformation* on the output error whenever it is fed back to the hidden nodes. This is by

(i) removing the nonlinear activation functions in the output nodes of MFN, and replacing them with the linear function $I(O_j)=O_j$,

(ii) connection weights to the output layer are kept to the initialized small random value, i.e., no learning is adopted for them. The random value is selected within a range [-x, x], where x is less that 1. The selected range is denoted by RIW.

(b) Two hidden layers H_{m1} and H_{m2} are adopted such that:

(i) both hidden layers are connected to the output layer,

(ii) the input layer is connected to H_{m1} only and H_{m1} is connected to H_{m2},

(iii) the tanh function is adopted for the hidden layers.

Figure 4.6 shows the general architecture for *GENE* approach. This approach will increase the probability of convergence to the global minimum, where the objective functions being minimized are constraints such that all weights / nodes are trained to reduce the network error. It has been shown in [66] that MFNs, which adopt linear activation functions in the output nodes, are universal approximations. With the linear activation function adopted at the output layer, fan-out the information from the network is another benefit. In addition, with no learning for the weights connected to the output layer, the learning problem is kept for the hidden layers only. Moreover, when using linear nodes for the input and output layers, the input-output information flow will not be distorting the training data, and hence the relationships between the input-output data can be easily retained.

4.5.5 Theoretical Basis of GENE Approach

In equation (4-10), d_{ik}^{l} contains the gradient w.r.t. the local node error. The network errors for the hidden and output neurons are given in (4-12), (4-13) respectively. Therefore the weights between the input and hidden layers are updated based on the local hidden node error. Generally, the major difference of the updating rule for the output layer and the hidden layer is in the evaluation of the node error. In the output layer, the error is a function of the desired and the actual output and the derivative of the neuron activation function. The objective function for the output layer is to minimize the network error. For the hidden layer the node errors are calculated on the basis of the transformed upper layer node errors. The weights connected to the output layer are updated every learning cycle. Hence, the transformation scheme for the network error to the hidden layer will also be

changing, and the subsequent minimization criteria for the hidden layers. The local error of the hidden nodes cannot be considered as a network error due to that change for its local objective function being minimized.

The *GENE* approach is developed to overcome the problem of node local error calculation for the hidden layers. A linear transfer function is used at the output layer with no learning performed at the output layer. The back propagation error is always transformed through the same initialized weight according to the RIW selected. Therefore the network error is back propagated to the hidden neurons such that the direction of the gradient does not change from the previous iteration. On the basis of the above assumptions, equation (4-13) is no longer valid for the output layer, and the network error is transferred to the hidden layer with the same transformation effect for every BP iteration (constraints by constant initialized weight). The convergence solution is now based on the search direction for equation (4-10) to minimize the transformed network error with a fixed transformation process for every application of an input.

The gradient search can be defined as $d_k \equiv -\nabla E(w^{(k)})$

With the linear activation function adopted for the output layer, the input and output layers are considered as fan-in and fan-out layers respectively. In this way, each hidden node is trained to optimize its local error such that the direction of the gradient for all of the nodes is always in the same direction as transformed from the network output error and towards the same origin.

The method of *GENE* approach can be applied with the configuration described earlier and can be used for the training of the data available at the start, as well as for any additional data that could be available after training with the original data; i.e., on-line

learning. This could be elaborated by considering a convex error surface, with a starting learning rate high enough to reduce the network error. The error is reduced with time and has a learning rate decay factor for smooth convergence to avoid abrupt changes when the error is close to a minimum. After a network is trained with the original data, the learning rate has been already reduced to a value such that the output error tolerance specified is reached. The reduced learning rate will be used for any additional training as required. Any further changes to the weights when the learning process starts again for additional data will be in the direction of reducing the network error and not reducing the local error, i.e., retaining the function approximation requirement. As both data sets are related to the same process, so the learning process will capture the function approximation of both data sets and not just interpolate each one at each training cycle. The learning rate in the second (additional) cycle is small, so that no abrupt changes to the weight will occur.

4.5.6 Network Error Flow and Weight Convergence Analysis

4.5.6.1 Standard Back Propagation (BP)

From equations (4-12) and (4-13) and for the first iteration step (i.e., $k=0$), with initialized weights $-0.05 < w^{l}_{jp}(0) < 0.05$, the relation $e^{l}_{j} << e^{L}_{j}$. holds. The evaluation of e^{l}_{j} has e^{L}_{j} as one of the multiplication terms, and all terms lie in the interval [-1,1]. The result of these multiplications will always be less than the smallest term used with the initialized weights. When the MFN is trained with BP or a variant and the coefficients are properly chosen, then the network error could be reduced during the training. The RMS values of the error for the output and hidden layers for a single hidden layered network are usually as shown in figures 4.7 and 4.8. In this case, $10e^{l}_{j} \geq e^{L}_{j}$ over the entire range. This extends the error relations for all k iterations. It can be concluded that *the error hyper-surface for the output layer is starting from a higher region than the error hyper-surface*

for the hidden layers at the start of the training and this relation remains for the entire learning process.

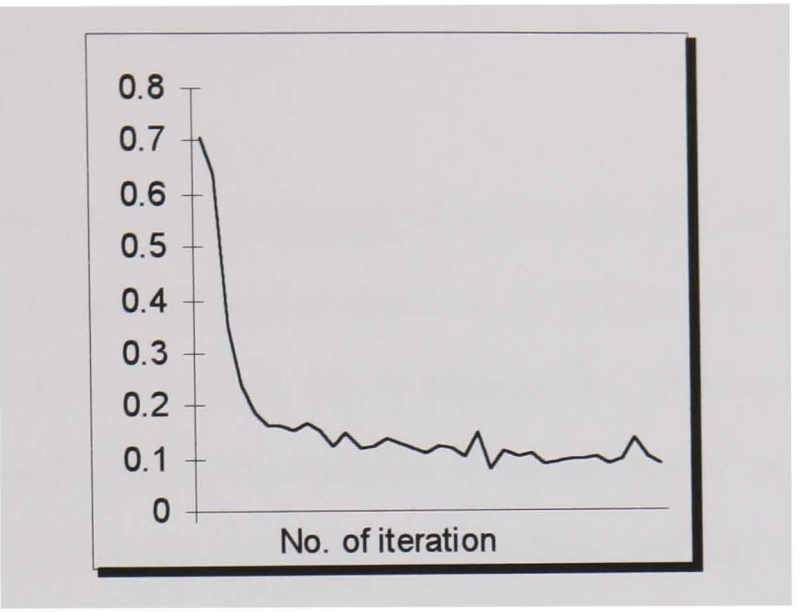


Figure 4.7: RMS error for output layer

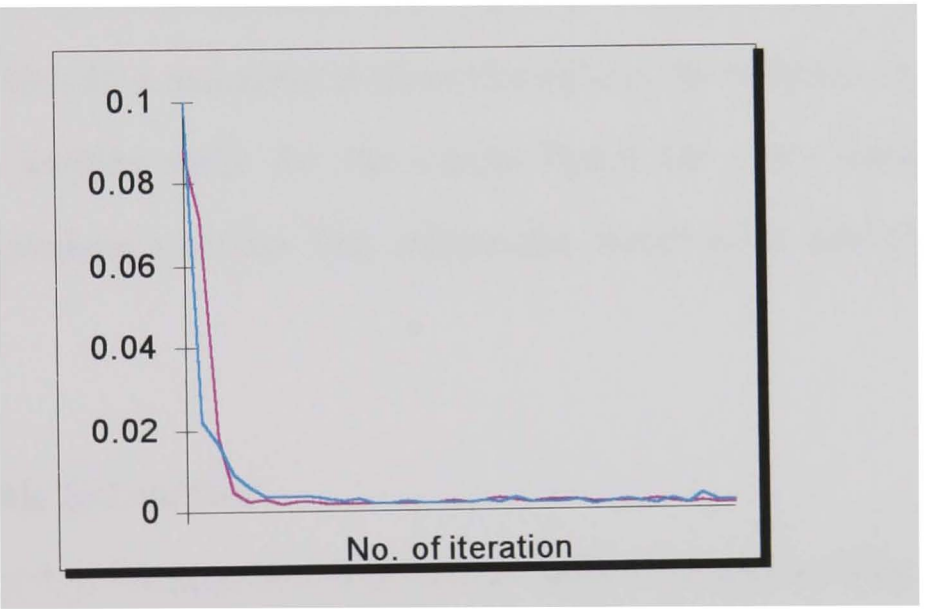


Figure 4.8: RMS error for Hidden layer

4.5.6.2 GENE Approach

The analysis of the dynamic behavior of the error flow throughout the network weights starts by rewriting the back propagation error equations (4-12) and (4-13) as:

$$e_j^l = g'^l_j \, SUM(e^{l+1}.w^l) \quad 1 \le l < L \qquad (4\text{-}15)$$

$$e_j^L = E.g'^L_j \qquad (4\text{-}16)$$

The error in equation (4-15) is composed of a multiplication of two terms, namely the derivative of the activation function and the summation of the upper layer error throughout the connection weights. By evaluating the summation for $N=1,2,..n$ with random error and weight values, the evolution of the summation surface can be thought as the scaled error throughout the connected weights. The resulting surface will be a smooth like surface regardless of the weight and error values. This is due to the fact that each multiplication term in the summation will result in a smaller value. This is similar to compressing the error to a smoother surface throughout the weights. By using the same (RIW) initialized weights (only for the output layer) for every iteration for the BP algorithm, it will produce a surface that reflects the output error directly fed back to the network.

4.5.6.3 Node Saturation

As shown in figure 4.9, the tanh function with its derivative approaching unity is therefore preferred to the sigmoid which has less than 0.3 derivative value. The preference is based on the fact that the network error is back propagated to the hidden nodes. The derivative is directly reflecting the node input. Using the tanh activation, the error evaluated by equation (4-13) will contain only the summation surface or a slightly scaled version, depending on the evaluation of the tanh derivative. Further comparisons between the tanh and the sigmoid activation are shown in figures 4.10 to 4.17, by using random error generated values < 1 for the errors with the number of neurons (N) is varied up to 90. The following cases of weight values are considered: (w=0.01, 0.03, 0.05, 0.1, 0.3, 0.5). It

can be seen that with the tanh activation it is easier to reach the saturation region as the number of nodes is increased and a higher value of weight is used. However for weight values less than 0.05, the activation is in the linear region and far from the saturation region. As shown in figures 4.14 to 4.17, the sigmoid activation will accelerate the node saturation problem. This confirms the use of the tanh function to avoid node saturation and the ultimate paralysis of the entire network.

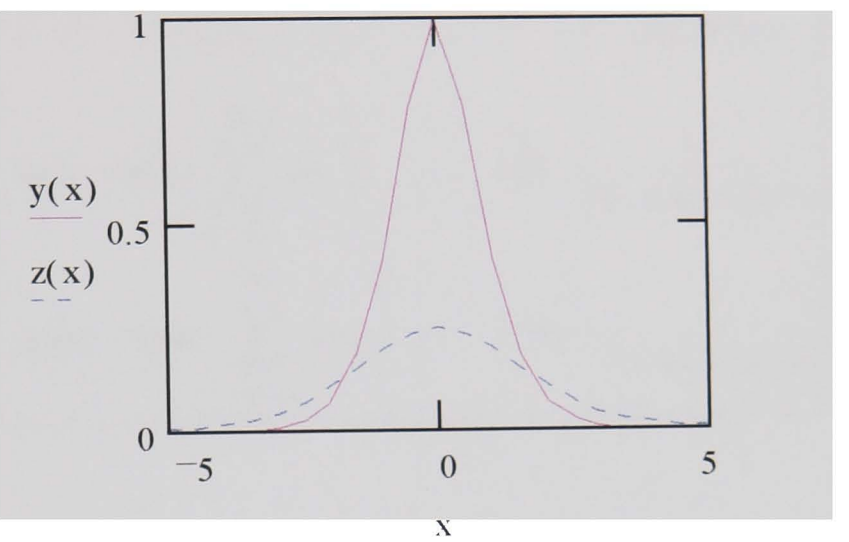


Figure 4.9: Comparison between the TANH " y(x)" and Sigmoid " z(x)" functions derivatives

4.6 Simulation Results

With the number of neurons (N) varied from 1 to 90, the initialized weight value is analyzed by considering the following cases for weight values:

$$w := .01 \quad , x := .03 \quad , y := .05 \quad , k := .1 \quad , \quad l := .3 \quad , \quad m := .5 \quad , \quad N := 1, 2 .. 90 \qquad (4\text{-}17)$$

Then the summation term f(.), the tanh node activation g(.) and the sigmoid node activation s(.) can be written for each case as follows:

$$f(w) := \sum_{i=1}^{N} w \cdot e_i \quad , \quad g(w) := \tanh\left(\sum_{i=1}^{N} w \cdot e_i\right) \quad , \quad s(w) := \frac{1}{1 + \exp(-f(w))}$$

$$f(x) := \sum_{i=1}^{N} x \cdot e_i \quad , \quad g(x) := \tanh\left(\sum_{i=1}^{N} x \cdot e_i\right) \quad , \quad s(x) := \frac{1}{1 + \exp(-f(x))}$$

$$f(y) := \sum_{i=1}^{N} y \cdot e_i \quad , \quad g(y) := \tanh\left(\sum_{i=1}^{N} y \cdot e_i\right) \quad , \quad s(y) := \frac{1}{1 + \exp(-f(y))}$$

$$f(k) := \sum_{i=1}^{N} k \cdot e_i \quad , \quad g(k) := \tanh\left(\sum_{i=1}^{N} k \cdot e_i\right) \quad , \quad s(k) := \frac{1}{1 + \exp(-f(k))}$$

$$f(l) := \sum_{i=1}^{N} l \cdot e_i \quad , \quad g(l) := \tanh\left(\sum_{i=1}^{N} l \cdot e_i\right) \quad , \quad s(l) := \frac{1}{1 + \exp(-f(l))} \qquad (4\text{-}18)$$

$$f(m) := \sum_{i=1}^{N} m \cdot e_i \quad , \quad g(m) := \tanh\left(\sum_{i=1}^{N} m \cdot e_i\right) \quad , \quad s(m) := \frac{1}{1 + \exp(-f(m))}$$

Now if p is the derivative of the sigmoid for k, l, m weight values,

j is the derivative of the sigmoid for w, x, y weight values,

h is the derivative of the tanh for k, l, m weight values,

i is the derivative of the tanh for w, x, y weight values,

Then the derivative equations for (4-17) can be written as follows:

$$p(k) := \frac{d}{dk} s(k) \quad , \quad h(k) := \frac{d}{dk} g(k) \quad , \quad i(w) := \frac{d}{dw} g(w) \quad , \quad j(w) := \frac{d}{dw} s(w)$$

$$p(l) := \frac{d}{dl} s(l) \quad , \quad h(l) := \frac{d}{dl} g(l) \quad , \quad i(x) := \frac{d}{dx} g(x) \quad , \quad j(x) := \frac{d}{dx} s(x) \qquad (4\text{-}19)$$

$$p(m) := \frac{d}{dm} s(m) \quad , \quad h(m) := \frac{d}{dm} g(m) \quad , \quad i(y) := \frac{d}{dy} g(y) \quad , \quad j(y) := \frac{d}{dy} s(y)$$

and the neuron error can be written as the summation multiplied by the derivative as follows:

$$\begin{array}{|llll|}\hline er(k) := p(k)\cdot f(k) & e(k) := h(k)\cdot f(k) & e(w) := i(w)\cdot f(w) & er(w) := j(w)\cdot f(w) \\ er(l) := p(l)\cdot f(l) & e(l) := h(l)\cdot f(l) & e(x) := i(x)\cdot f(x) & er(x) := j(x)\cdot f(x) \\ er(m) := p(m)\cdot f(m) & e(m) := h(m)\cdot f(m) & e(y) := i(y)\cdot f(y) & er(y) := j(y)\cdot f(y) \\ \hline \end{array} \quad (4\text{-}20)$$

A)Tanh activation analysis: Using a random generated error values < 1; and for different cases of weight values: w=0.01, x=0.03, y=0.05, k=0.1, l=0.3, m=0.5 and using the number of neurons up to 100 (N); the node activation is found to reach the saturation region as the number of nodes is increasing as well as with the higher values of weights, as shown in the figures below. However, for weight values < 0.05 the activation is in the linear region and far from the saturation region.

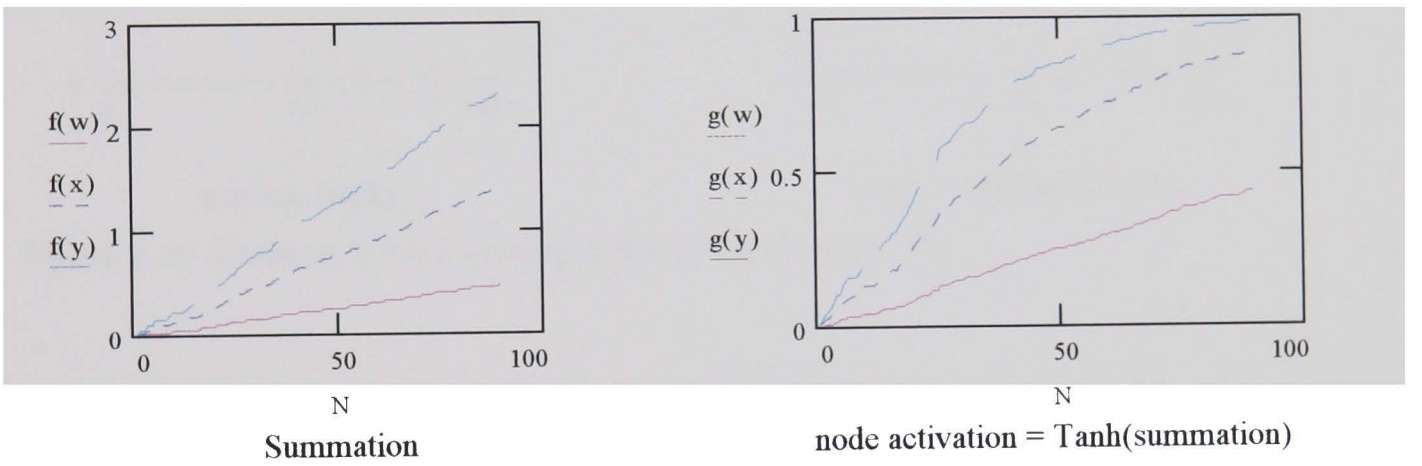


Figure 4.10: Summation & node activation for the TANH function - I

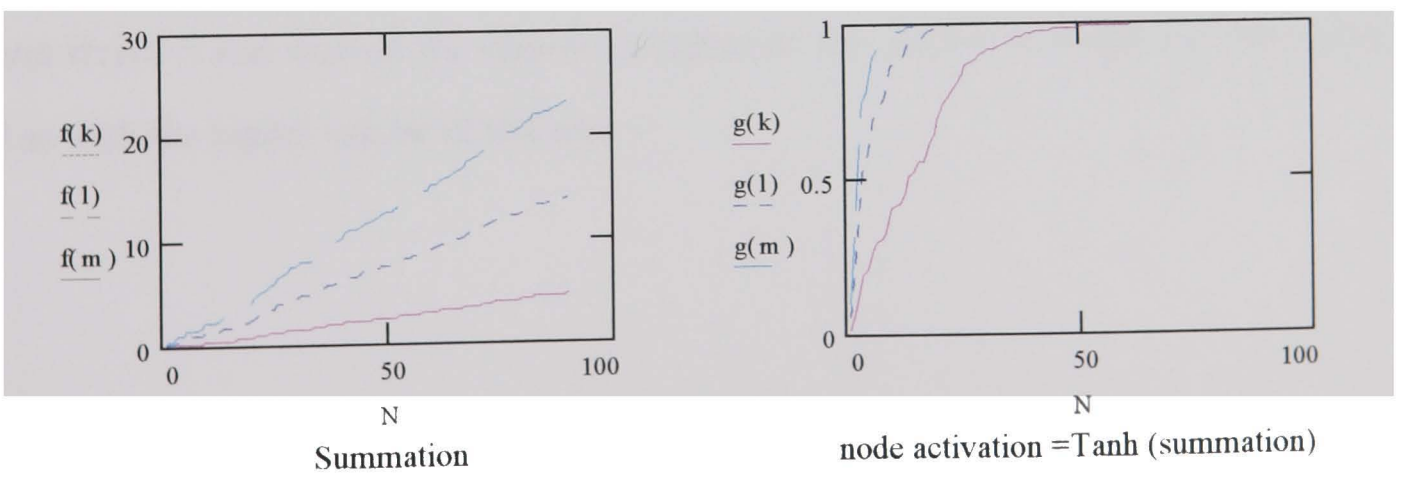
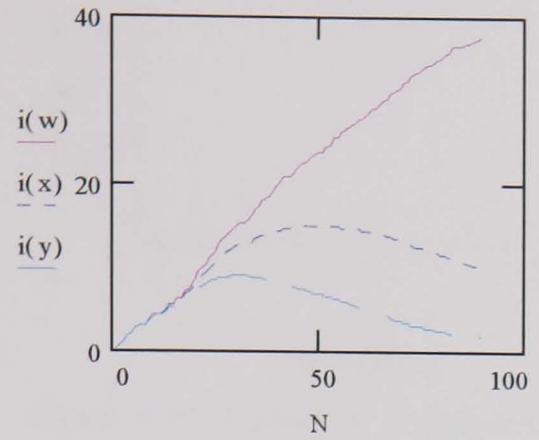
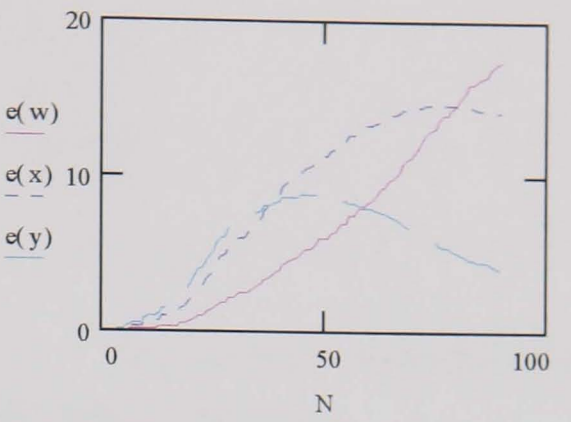


Figure 4.11: Summation & node activation for the TANH function - II

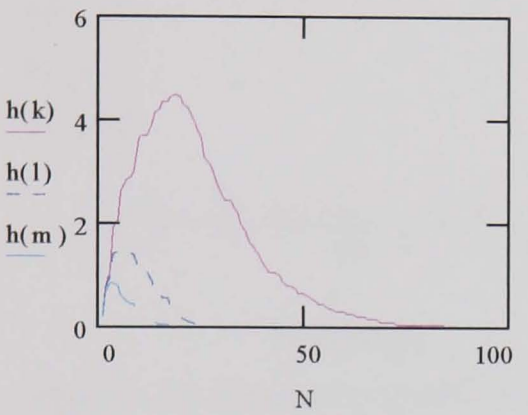


gradient (tanh)

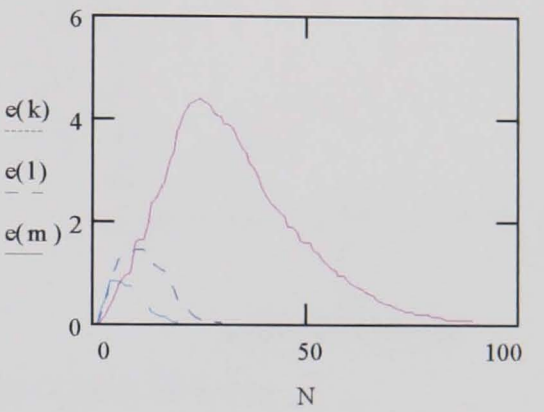


error = gradient x (sum)

Figure 4.12: Gradient & error values for the TANH function - I



gradient (tanh)



error = gradient x (sum)

Figure 4.13: Gradient & error values for the TANH function - II

B) <u>Sigmoid activation analysis</u>: Using a random generated error values less than 1; and for different cases of weight values : w=0.01, x=0.03, y=0.05, k=0.1, l=0.3, m=0.5 and using the number of neurons up to 100 (N); it is found that the node activation is always greater than 0.5 and reaches the saturation region as the number of nodes are increasing as well as with the higher values of weights.

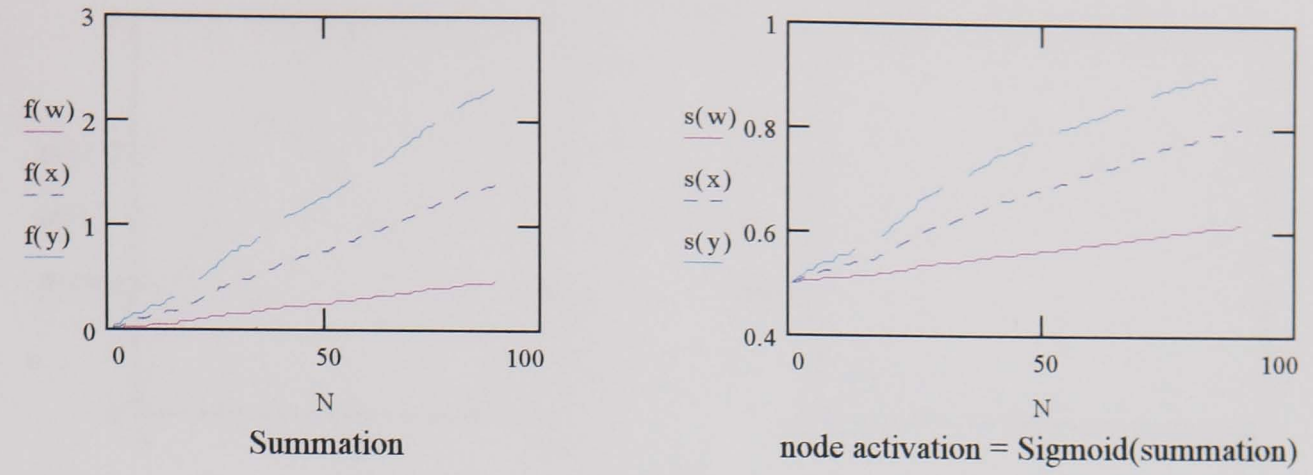


Figure 4.14: Summation & node activation for the Sigmoid function - I

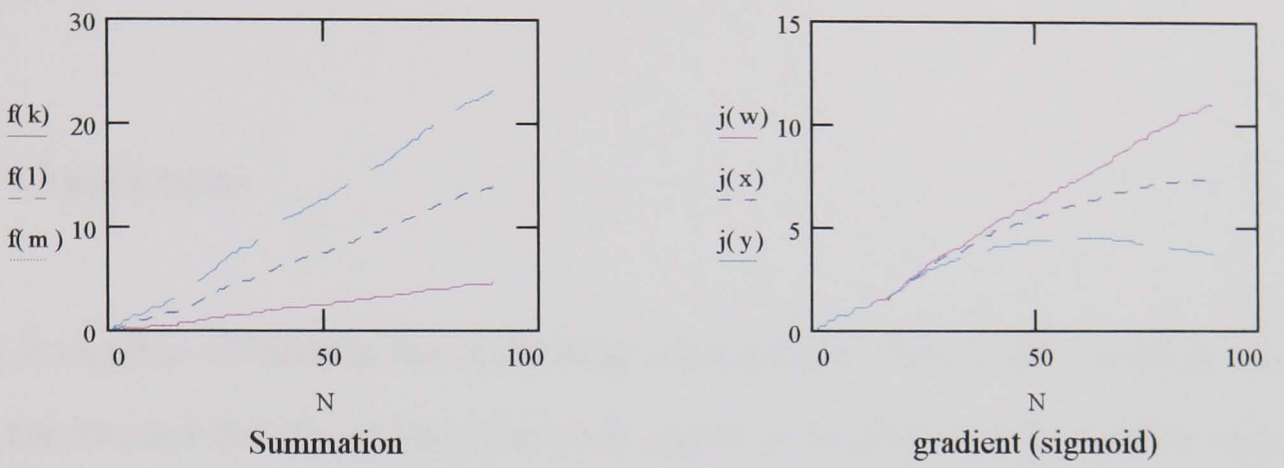


Figure 4.15: Summation & node activation for the Sigmoid function - II

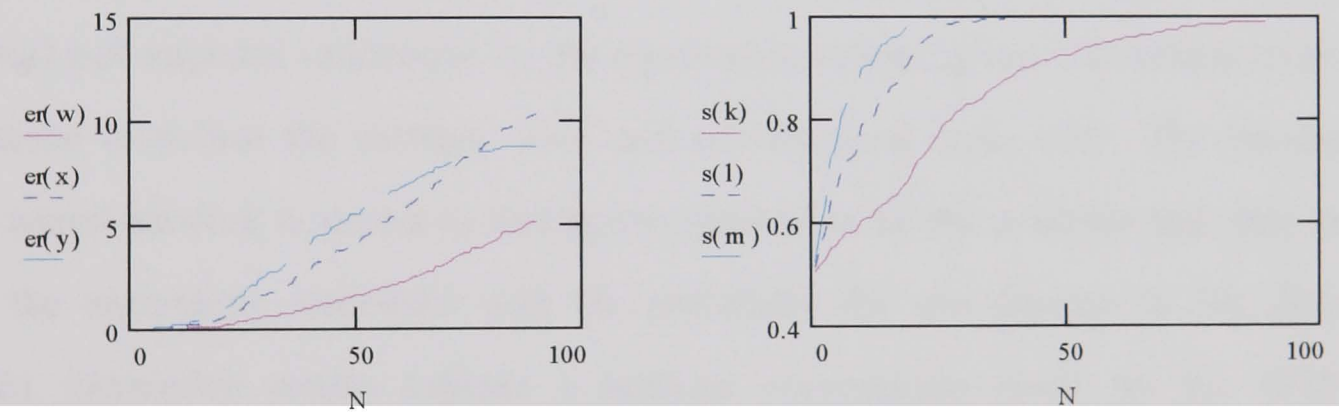
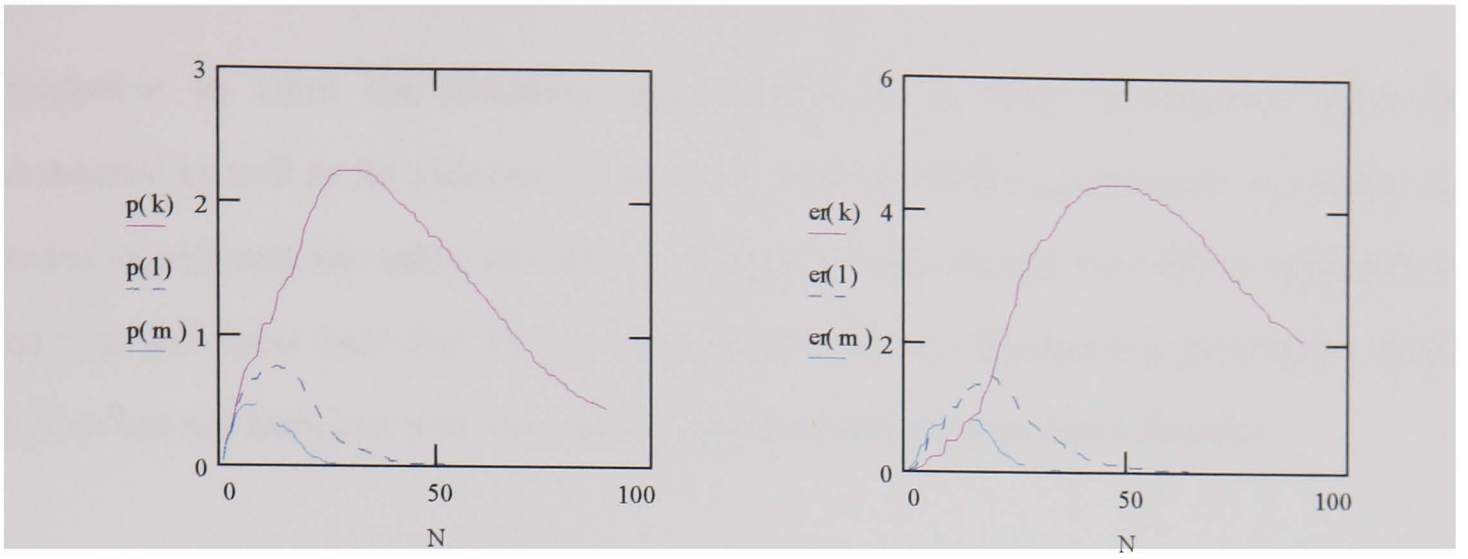


Figure 4.16: Gradient & error values for the Sigmoid function - I



gradient (sigmoid) error = gradient x (sum)

Figure 4.17: Gradient & error values for the Sigmoid function - II

4.7 Conclusions

BP net is capable of learning the underlying relationships. Once a BP model is available, then it can be used directly on line. The weak points of an ANN are the number of training examples required for generalization is not known and the long training time. The *GENE* approach for improving the learning capability of the BP algorithm is developed. Both the theoretical and empirical validations for the approach have been given. The hidden layer is only trained to reduce the network error and not the local node error. The resulting trained neural network is shown to intelligently generalize all the available test data and reduce the uncertainty associated with the probability for convergence to the global minimum. Simulation results indicate a superior convergence speed for the *GENE* approach and robustness for analog data. Robustness of the network arises when the network is trained with new data and a practical example is presented in chapter 6. The network will not forget the previously learned data in order to learn the new data. Other benefit is the increase in the type of the function approximation such that highly nonlinear functions could be predicted. This is demonstrated by predicting accurate brine levels for MSF plants in chapter 5. In the following chapter the capability of the standard back

propagation to learn the function approximation of a MSF Desalination plant is investigated as well as for supervisory set point control function generation. In chapter 6, detailed experiment for using the back propagation that utilized the GENE approach is described and show how that by providing additional data training the previously learnt function has not been lost with the capability to learn the new required function.

Chapter 5 Application of Modelling & Control of MSF Plants by ANN - I

5.1 Introduction

As described in chapter 2, accurate model and control of MSF desalination industry is an important subject for solving the numerous problems that arise in operation and affect the plant stability. It is well known that imprecision, non-linearity, large size and complexity frequently characterize MSF desalination modelling solutions. They are based on empirical equations obtained from laboratory work and there is no obvious expression to calculate the complete model. For the purpose of effective control of the brine level inside the stages, it would be of great significance to obtain accurate model for it. Recently, ANN has been proposed to solve MSF modelling and identification problems [5], [67], [68]. Compared with other methods, the outstanding merits of an ANN are:

1. The speed of calculation is high, once the network is properly trained. The computational power involved in calculating the output does not depend on the complexity of the process and/or the output precision requested, it depends only on the Neural Network specific architecture and it is, in general, low. Although most ANN applications are still simulated on digital computers rather than being implemented on hardware, there is still significant saving in computer time in comparison with numerical simulation and other methods.

2. There is no restriction on modelling, in contrast to many other methods. Numerical simulation can be used to generate training and testing examples, or can be obtained from the real plant during the commissioning stage of the plant. Thus the network model will behave exactly like the plant which the training data was extracted from.

3. With the low computational power involved in computing the output, extensive simulation analysis can be a very useful tool for the operator and the engineers (e.g.: to implement a what-if-analysis).

An ANN is therefore a potential candidate for an on line adaptive control and fault diagnosis.

This chapter presents a method of applying ANN to reproduce the process behavior during load variations of MSF desalination plant which can be used for simulation, modelling and control purposes. The aim is to demonstrate the use of ANN based on input-output data as viable alternative to mathematical models. The inputs to the network model are the manipulated variables (set points to regulatory control) and the outputs are the controlled variables plus the vapor temperatures in each flash stage.

The performance of the network model is studied by comparing the behavior of the network to an actual behavior of the plant, due to an increase or decrease for one of the manipulated variables. In contrast to the approach of Parenti, Bogi and Massarani where data is obtained from a detailed process simulator [67], in this study the data are obtained from a commercial MSF plant. The data obtained covers load increase and decrease between 60% and 100% production load. These data are used for training the network as well as for studying the network performance for making prediction with particular

attention to the brine level in the first stage. In this study only one network is used to model the process behavior in contrast to the cross bar structure used by Parenti, Bogi and Massarani [67] where several networks are deployed such that each one is used to estimate one single output variable. Using one model is expected to capture all functional relations between all variables.

5.2 Review of the Literature

El-Hawary [5] has studied the possible applications of neural networks (based on the application of the same) to the power system. To the best of our knowledge, the first attempt for applying ANN to model MSF plants was explored by us [39], [40], [41]. An ANN with one hidden layer was trained using the BP to learn a set of input-output data during load variation. Ramasamy and others [68] also used ANN with one hidden layer for modelling the relationships between the input-output variables. For a (9-5-3) network, they have considered the controlled variables to be the brine levels of the first and last stages, and the product rate as an output for the model. They suggested the manipulated variables to be:

- Brine recirculation flow rate Brec
- Top brine temperature TBT
- Sea Water inlet flow rate SWin-F
- Sea water temperature SWt
- Sea water circulation flow rate SWrec-F
- Makeup sea water flow rate MAKEUP
- Brine blow down flow rate. BBD
- Steam input flow rate ST-F

For 18 stage MSF desalination plant, a fully connected MFN with one hidden layer and using the sigmoid transfer function for the neurons was considered. The data was obtained by conducting designed pseudo random binary sequence (PRBS) tests on a dynamic simulator model of MSF plant for about 2100 intervals. To reproduce the process behavior, they used the current value of all the manipulated variables and the controlled variables as neural network inputs to predict the outputs (controlled variables) at the next instant. The input-output data were divided into three different sets for training, cross validation and testing of the neural network model. Each data set contained 700 data patterns. Satisfactory network performances were obtained by training the network with BP. However, the result based on the use of PRBS tests to obtain the data from the simulator does not provide the necessary validation. It is necessary to test on practical plant data over the entire operating range, including the nonlinear regions, and carry out rigorous simulations. The work in this chapter is covering this drawback.

Parenti, Bogi and Massarani, have used ANN with one hidden layer for modelling the MSF process based on an ANN steady state estimator, which has been trained on data collected from a process simulator. They suggested the following inputs [67]:

- Evaporating brine inflow
- Evaporating brine temperature
- Evaporating brine salinity
- Sea Water temperature

The outputs from the network after training are the value of the following 8 variables for each steady state record data base:

- Brine heater steam flow
- Makeup flow
- Blow down flow

- Evaporating brine out flow
- Evaporating brine out temperature
- Evaporating brine out salinity
- Gained output ratio
- Distillate production

They have not considered the brine levels as one of the controlled output. It appears they have difficulty in training such a network, and suggested splitting the problem into 8 networks each have 4 inputs and one output. They used the batch learning with an epoch size of 16 examples, with 5000 updating cycles for every net.

5.3 Constraints of the Study

The use of one ANN to predict a practical comprehensive range of operating conditions including the brine level has not yet been shown to be feasible and studies have been confined to restricted situations. The nature of restrictions imposed have a very strong influence on the success of an ANN in predicting the process behavior during load variations (load trajectories) or on the types of inputs or outputs required. Data was obtained from MSF plant at Al-Taweelah consists of 16 stages (13 recovery and 3 reject stages). The conditions of the study were as follows:

1. Maximum distillate product load (100 %) corresponds to 1350 t/h and minimum distillate product load (60%) corresponds to 650 t/h.
2. LP steam temperature is 182.5 degree C, and steam pressure of 0.75 bar for all data.
3. Two network topologies are used. When using the BP learning algorithm, a single hidden layer is adopted, while two hidden layers are adopted when using the GENE approach.

Examples were generated from the plant during load increase and decrease between 60% and 100% production load. This was done twice so that two different trajectories are obtained.

5.4 ANN Considerations

This application falls into the category of a nonlinear mapping between inputs and outputs. The ANN architecture adopted here is a MFN with one hidden layer using a Tanh activation function when using the BP learning algorithm, while two hidden layers are adopted when using the GENE approach.

5.4.1 Choice of ANN Inputs

The choice of inputs is an extremely important factor in the successful use of an ANN. The modelling problem here is to find suitable manipulated variables, which affects the controlled variables. For the plant model, having the goal to reproduce the process behavior from the data obtained, the inputs to the neural network model (NNM) are the process manipulated variables (set points to regulatory control) and the outputs are the corresponding controlled variables. Moreover, by having the vapor temperatures as additional output from the network model, it will provide information on all the flashing stages and verifies the modelling capability of ANN for reproducing the process behavior.

The problem connected with the control of both brine recirculation loop and the top brine temperature loop is mainly due to the interaction between the two loops and the process directly. The control problem can be hardly solved with the tools of the classical and advanced linear control theory because of the mismatch between the real process and the model on which based upon. A major source of the mismatch is the nonlinear characteristic of the process. Load variation for the distillate production rate, requires

generation of pre-determined steps to the set points of the regulated variables for the two loops. The product quantity can be maintained with different level patterns in the stages. A bounded constraint is desirable to keep the brine level within the upper and lower level limits in each stage. Preventing the brine being flooded with the vapor to the condensate tray affecting the purity of the product is by the upper bound, while the lower one is for preventing the blow through of the brine at the inter stage to cause loss of the differential pressure between stages. Increasing the brine recirculation flow will increase the amount of brine to the 1st stage, but tend to decrease the level in the last stage. In general an increase in the brine level in stages decreases the efficiency. A level control with the brine blow down flow as the manipulated variable for the last stage is usually employed. However no control can be done on the first stage and any other inter-stage levels. Difficulties in classical approach can be summarized as follows:

(1) The evaporation process is an open-loop unstable system, because of the existing two-phase flow mixture.

(2) There is an explicit limitation on the magnitude of the control signal that can be used for control.

(3) Plant parameter uncertainties and of nonlinear phenomena which cannot be modelled.

Therefore effective control of the brine levels is to obtain accurate models for them. Attempts in the literature appear using steady state analytical models. An alternative is the empirical process model for the brine level to be used in adaptive control and fault diagnosis of MSF. Thus the inputs to the NNM network model are selected as:

- Brine recirculation flow rate Brec
- Top brine temperature TBT
- Sea Water inlet flow rate SWin-F
- Sea water temperature SWt
- Sea water circulation flow rate Swrec-F

- Makeup sea water flow rate — MAKEUP
- Brine blow down flow rate. — BBD
- Steam input flow rate — ST-F

The outputs of the model are

- Brine Level in first stage — BL-1
- Brine Level in last stage — BL-L
- Distillate product flow rate — PROD
- Distillate Level in last stage — DL-L
- Brine to heater temperature — BTHT
- Seawater from reject section temperature. — SWrej-t
- Vapor temperature in all stages (16 no.'s) — VT-1...VT-16

As an alternative to the TDL for the past values, each input to the network model has two associated vectors $v_1(k)$, $v_2(k)$ and are defined as follows:

$$v_1(k) = u(k) - u(k-1)$$

$$v_2(k) = \sum_{n=0}^{3} v_1(k-n)$$

Where $u(k)$ is the input at instant k. Thus, v_1 represents the change in the current sample from the previous one, while v_2 will represent the accumulated change for the current and the past 3 samples. Thus the model will have 8x3 inputs instead of 8x6 inputs. This reduces the number of inputs to the network, thus reducing the size of the input vector and hence the number of free parameters (weights and thresholds) in the network to be determined during training. This is important for any non-parametric technique since the number of training examples required grows exponentially with the dimension of the input vector. A rule of thumb which is sometimes cited is that the number of training examples should be of the order of ten times the number of weights (see Section 4.2).

5.4.2 Data Collection, Preparation and Preprocessing (Scaling)

As the model required to handle analog data, the acquisition hardware and software used have the capacity of 1282 analog input values that can be measured with a scanning cycle of 2 seconds and the integration conversion method is used to read the coded digitized values (12 bit + sign). MSF plant can be characterized as a relatively slow process. Tracking one particle of brine from the recycle start point at the last stage, it would require few minutes to complete the cycle reaching the last stage again. Considering tracking one particle of brine inside the brine recycle loop, the 2 seconds is a very low scanning cycle, so a printout of the data has been implemented every 30 seconds. Data was obtained during 60% to 100% load variation of a MSF plant (13 recovery stages and 3 reject stages) for Al-Taweelah-A desalination plant which is equipped with a process computer system.

As the network nodes transfer function used in this study is the hyperbolic tangent, it is required to map the data into the working range of such transfer function, which lies between -1.0 to +1.0. For the software used (Neural Ware Professional II plus), one can specify the maximum and minimum ranges within which the real input-output data lies. A linear mapping is then performed on the real data to the network input-output. By this way the distribution of the data could be easily specified to cover the normal distribution of each measurement. For example, the distillate flow that could be produced from the plant under study is in the range of (650 - 1350 t/h), and these were used as the minimum and maximum values respectively. For the brine recirculation flow the maximum value is specified by the pump capacity used for recirculation. The minimum value is the minimum allowable velocity in the condenser tubes to avoid excessive fouling and erosion. For the top brine temperature, the minimum and maximum values used are the corresponding

values to the minimum and maximum values of the distillate flow. It was found that it is very important to set the minimum and maximum values correctly, otherwise the network may never learn and the error function may be of high values and cannot be reduced, particularly if the minimum and maximum values for the different measurements are inconsistent. On the other hand if the minimum and maximum range is enlarged, the resultant error after training will relatively larger (e.g.: 1% error of the range 0-1500 t/h is 15 t/h, while for a range 650-1350 t/h is 7 t/h). Other advantage from specifying the maximum/minimum values is that it can be considered as an embedded constraint that can be utilized for fault diagnostics. The maximum and minimum values used during the training in this study are shown in Table 5-1.

5.4.3 Training Consideration

The connection weights and thresholds of the NNM were adjusted by training using the GENE approach to examine its performance for prediction of the brine level. This is compared to the performance when the ANN is trained using the BP based on the normalized cumulative delta rule. Each weight and threshold were assigned an initial random value between -0.05 and +0.05. The learning rate, momentum and temperature values for the output and hidden layers are selected as shown in Table 5.2. The epoch size chosen is 16.

5.5 Empirical Process Model Development

5.5.1 BP Approach

In order to study the ANN approach using the BP algorithm for training for MSF plant modelling during load change, the network has to be trained to predict the correct values of the controlled variables for every record of data obtained. For a single hidden layer with

20 hidden nodes and by using the back propagation learning algorithm it was found that adequate learning for the required function approximation has been achieved. The data obtained contains 1160 records. For the training phase, the data is divided into two parts; each one covers a complete load increase and decrease. The training is conducted using the first part for 500 iterations, followed by additional training using the second part and then finally the training is conducted using the whole data. Figure 5.2 shows a good approximation has been achieved for the various temperatures and flow variables. However, the results in figure 5.1 for the brine level of first stage shows non conformance with the actual data. The test result does not follow the oscillatory characteristic of the actual brine level in the first stage, but a smooth curve has been produced. As will be shown later, using GENE approach for learning does not have such drawbacks in estimating the oscillatory characteristic for the brine levels over the entire range of data obtained. It is worth to note that the low effort and the computational power required for obtaining the result as compared to any other mathematical formulation for the MSF modelling problem.

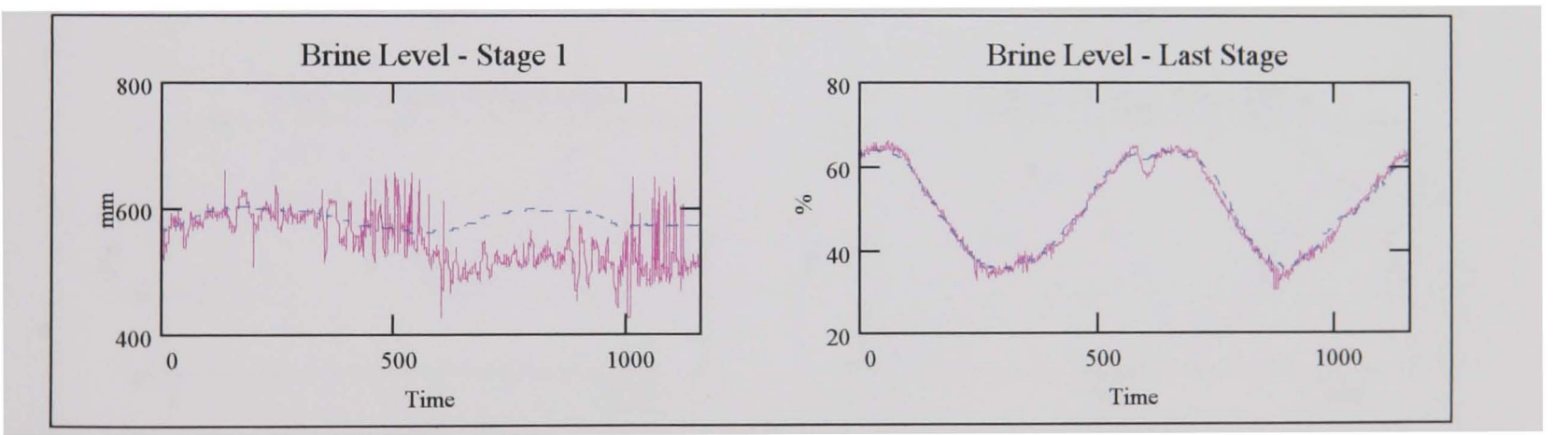


Figure 5.1: Test results for modelling MSF plant brine levels (1st and last stage) using BP learning algorithm and 20 hidden neurons.

The dotted lines represent the network output and the lines represent results from the actual plant.

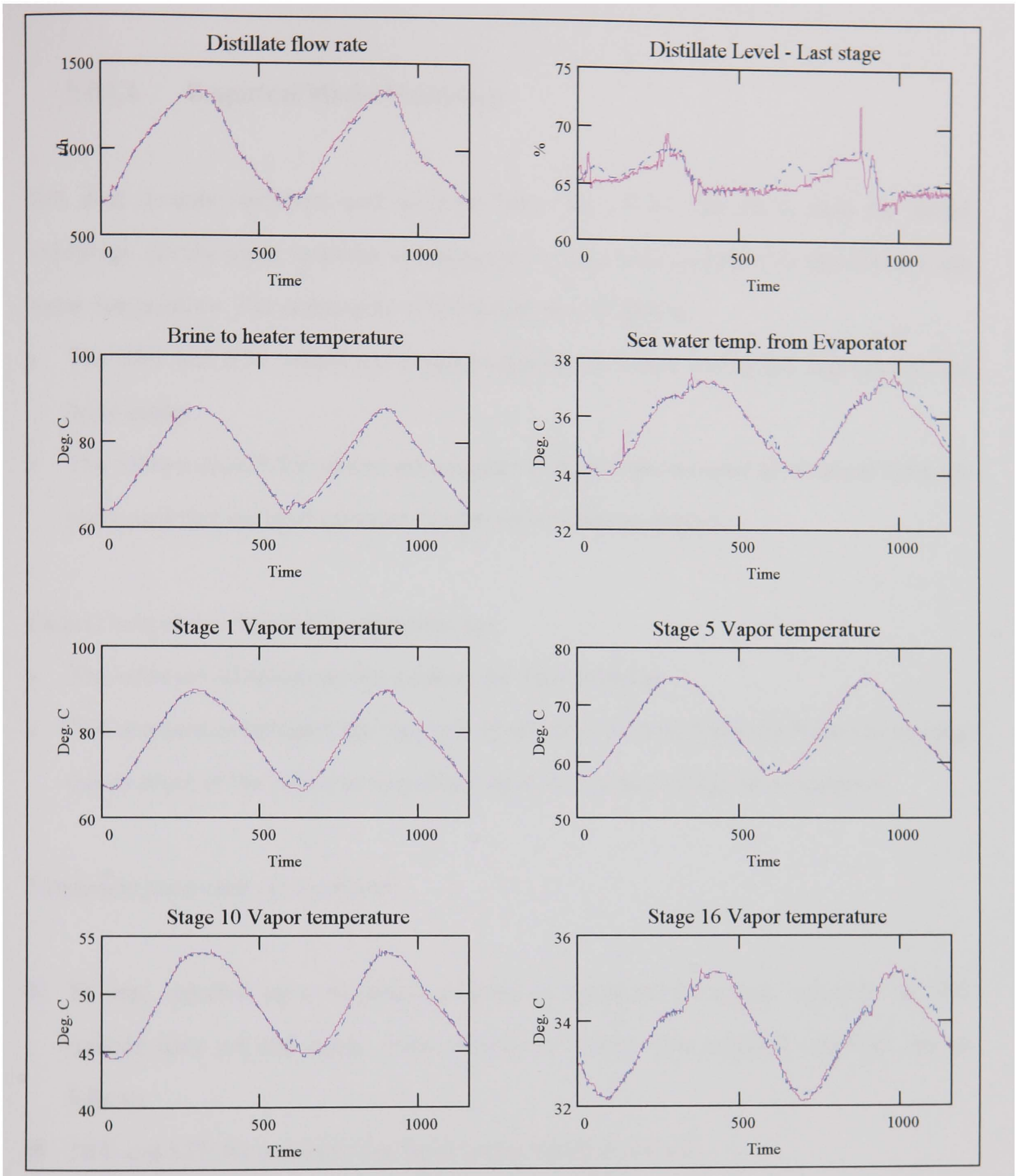


Figure 5.2: Test results for modelling MSF plant with 16 stages using BP learning algorithm and 20 hidden neurons.

The dotted lines represent the network output and the lines represent result from the actual plant.

5.5.1.1 Empirical Model Validation

100 data records represent load increase from 700 t/h to 960 t/h is used for model validation. All the input variables obtained are varying with exception to the constant sea water temperature. The constraints of the model are as follows:

- The TBT and STF values are coupled together to follow the logical sequence of the brine heater.
- The SWrec and SWFin values are coupled together and are used as obtained from the plant such that the inlet seawater temperature remains constant.

Factors considered for model validations are:

- The selection of hidden nodes numbers for such mapping.
- The consistence between the network model and the actual process behavior to keep one or more of the inputs to a constant value for a selected region of operation.

Simulation tests used are as follow:

1- **T1 test**: selected input variables are kept to a minimum constant value for all 100 records data set and apply these records to NNM. The selected variables are as follows:

- TBT and STF for constant heat input to the desalination unit.
- Makeup flow and brine recycle flow. The seawater recirculation flow, brine blow down and seawater inlet flow are the only varying inputs.

It is envisaged no variation for the product flow under this test.

2- T2 test: same as T1 but using the maximum values found in the data set.

The minimum and maximum values found in the data set are shown in table 5.3.

When an input variable is constant, the associated vectors v_1, v_2 will have a zero value for all k's. Simulation results are shown in figures 5.3, 5.4, 5.5, and 5.6 for NNM when using 4, 12, 20 and 28 hidden nodes respectively. Each figure includes 6 results for each output. The response for each output is described in the following:

- **Brine level cell 1** :

1- MMN with 4 hidden nodes: For the first 70 records, the level response follows a constant value for T2, while for T1 the level follows an oscillatory response as shown in figure 5.3 (a). The level response for T2 is lower than that for T1.

2- MMN with 12 hidden nodes: the level follows the same oscillatory response of the original data set but at a constant mean high value for T1 as shown in figure 5.4(a). Similarly, the response for T2 is the same but with a constant mean low value.

3- MMN with 20 hidden nodes: the level follow the same oscillatory response of the original data set but at a constant mean high value for T1 as shown in figure 5.5(a). Similarly, the response for T2 is the same but with a constant mean low value.

4- The response for MMN with 28 hidden nodes is in a reverse manner when compared to MMN with 4 hidden nodes. The flow response for T1 is lower than that for T2. While visa versa for MMN with 4 hidden nodes.

- **Brine level cell 16** :

5- MMN with 4 hidden nodes: The level response follows a constant value for T1, while for T2 the level follows an oscillatory response as shown in figure 5.3 (b).

6- MMN with 12 hidden nodes: the level follows a constant value for T1 and T2 as shown in figures 5.4(b).

7- MMN with 20 hidden nodes: the level follows a constant value for T1 and T2 as shown in figures 5.5(b).

8- The response for MMN with 28 hidden nodes is similar to that for MMN with 4 hidden nodes, but in the reverse manner. The oscillatory response for the level for T2 is shown in figure 5.6 (b).

- **Product flow** :

9- MMN with 4 hidden nodes: The flow response follows a constant value for T2, while for T1 the flow follows an increasing response as shown in figure 5.3 (c).

10- MMN with 12 hidden nodes: the flow follows a constant value for T1 and T2 as shown in figures 5.4(c).

11- MMN with 20 hidden nodes: the flow follows a constant value for T1 and T2 as shown in figures 5.5(c).

12- The response for MMN with 28 hidden nodes is similar to that for MMN with 4 hidden nodes, but in the reverse manner. The increasing response for the flow for T1 is shown in figure 5.6 (c).

- **Sea water reject temperature**: the variation for this variable using T1 and T2 data set as shown in all figures is due to the variation of the sea water recirculation flow and the sea water inlet flow. But again the response for MMN using 4 hidden nodes is reverse to the response when 28 hidden nodes were used.

- **Brine temperature to brine heater**: same response as the product flow.

- **Vapor temperature cell 1**: same response as the product flow.

From the above MMN with 20 or 12 hidden nodes, it shows that better performance is achieved compared with that of 4 or 28 nodes in the hidden layer in simulating the process behavior. This model can be utilized in case where the brine level of the first stage is not an important factor. Figure 5.7 shows the response of the network could not follow the oscillatory response for the brine level in the first stage where different number of hidden nodes is used. However, in the following section, the GENE approach is explored for prediction of accurate brine stage level.

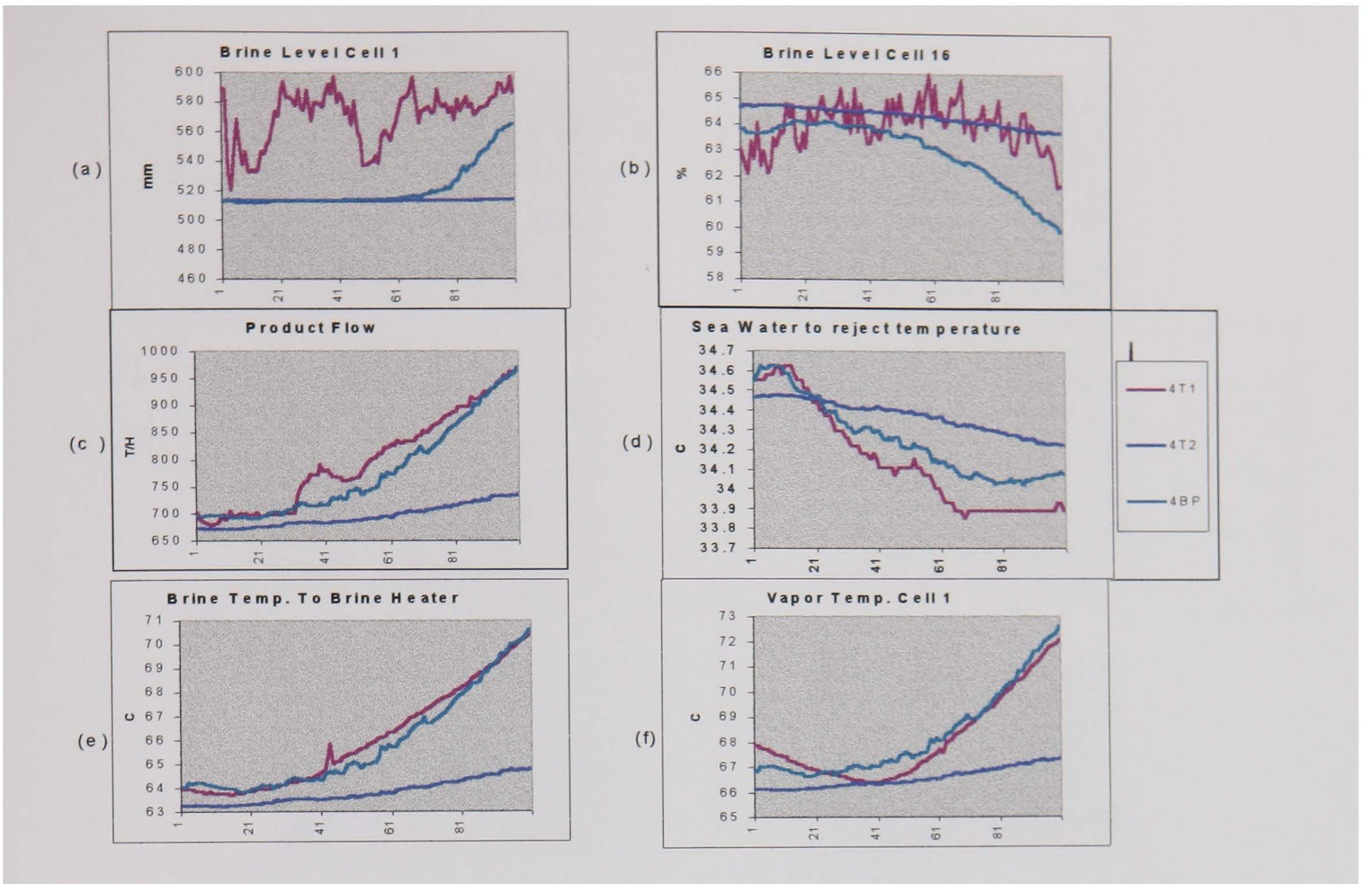


Figure 5.3: Network model simulation results using T1and T2 data after training by BP algorithm with 4 hidden nodes.

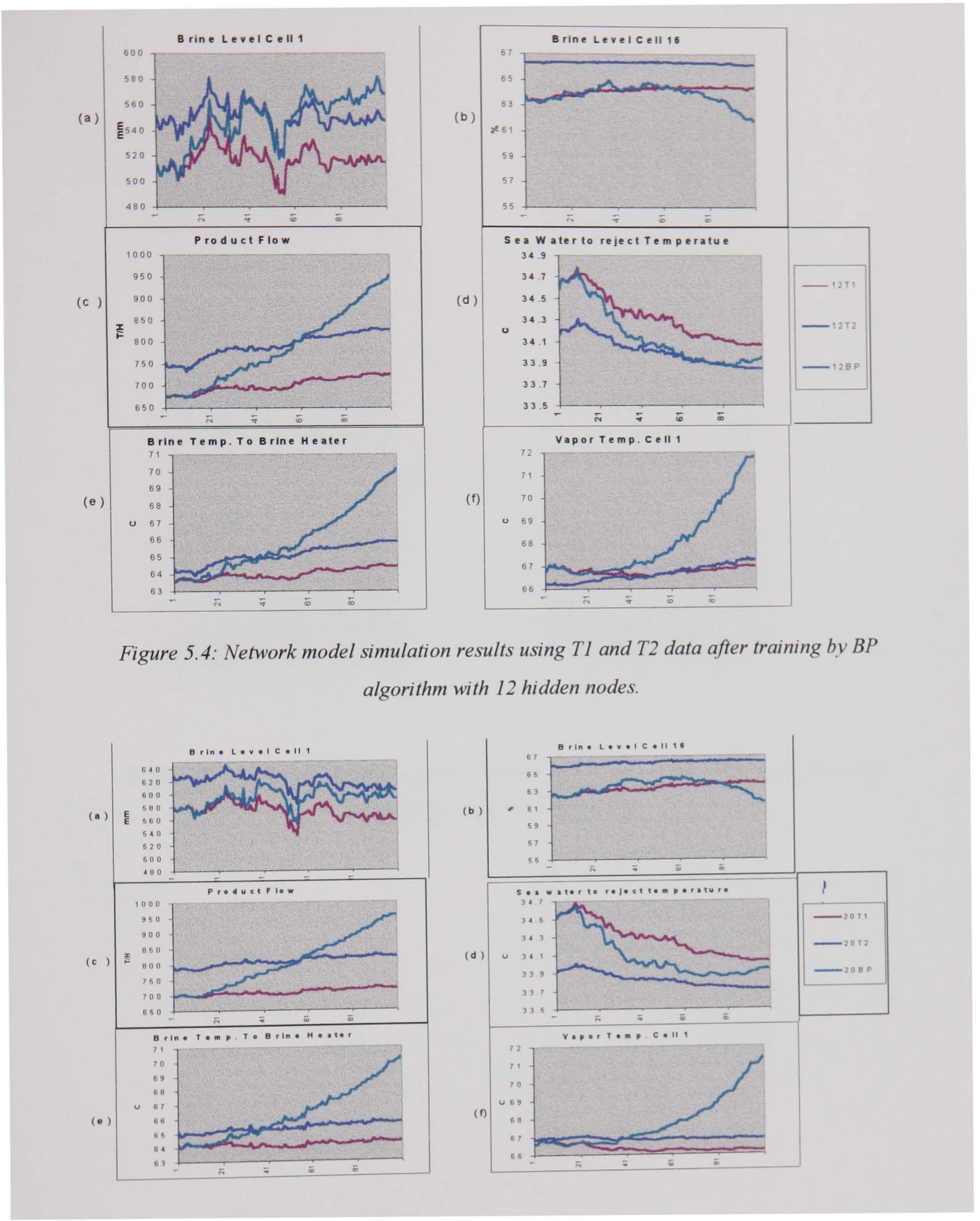


Figure 5.4: Network model simulation results using T1 and T2 data after training by BP algorithm with 12 hidden nodes.

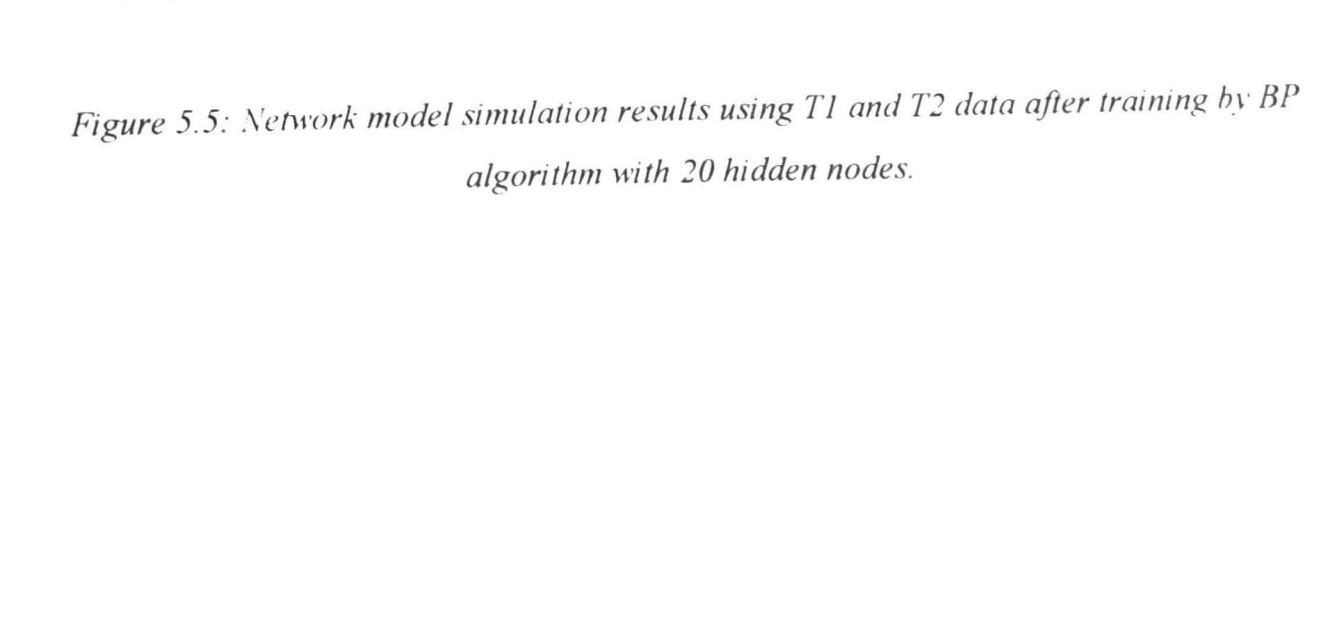


Figure 5.5: Network model simulation results using T1 and T2 data after training by BP algorithm with 20 hidden nodes.

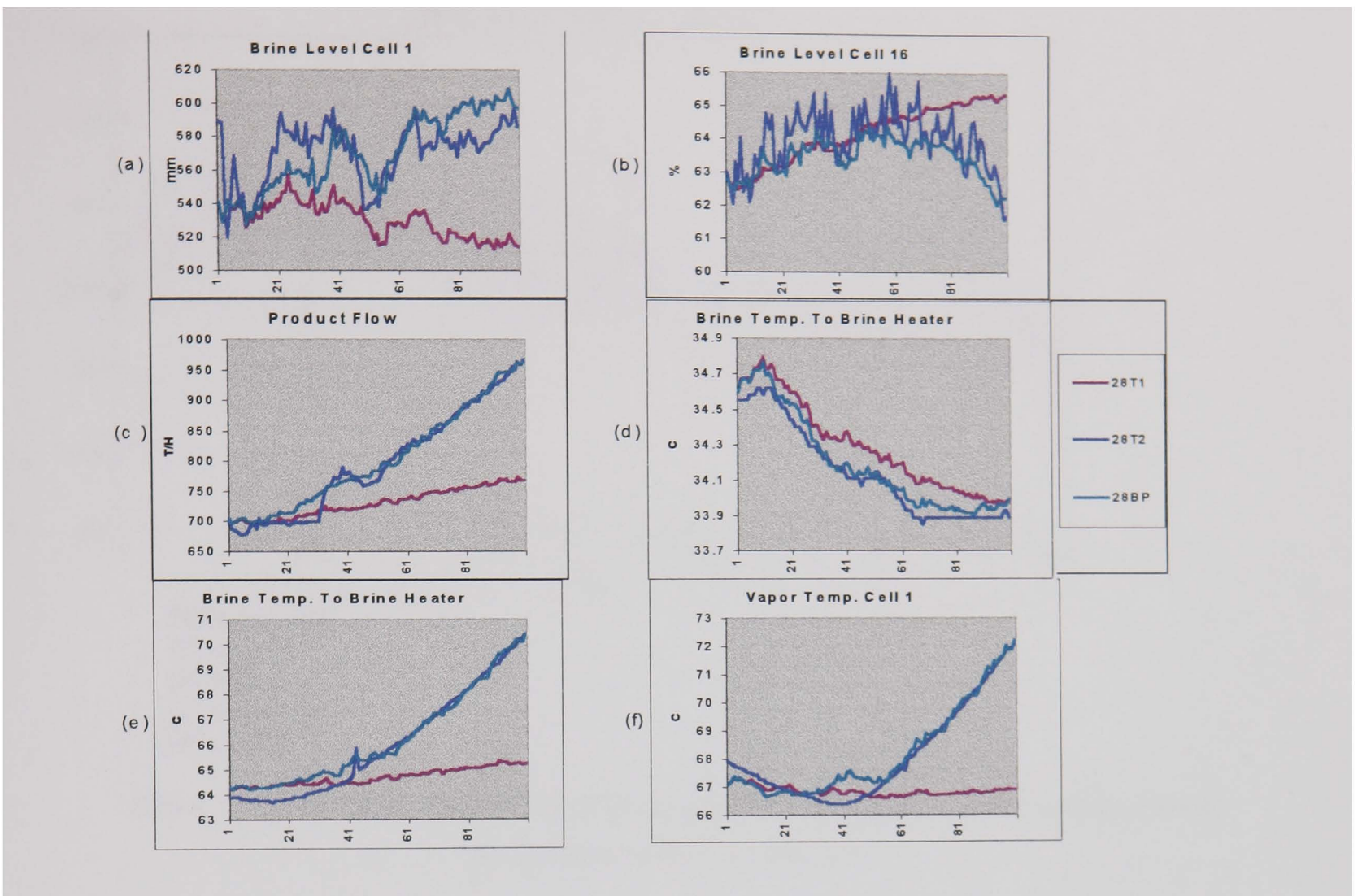


Figure 5.6: Network model simulation results using T1 and T2 data after training by BP algorithm with 28 hidden nodes.

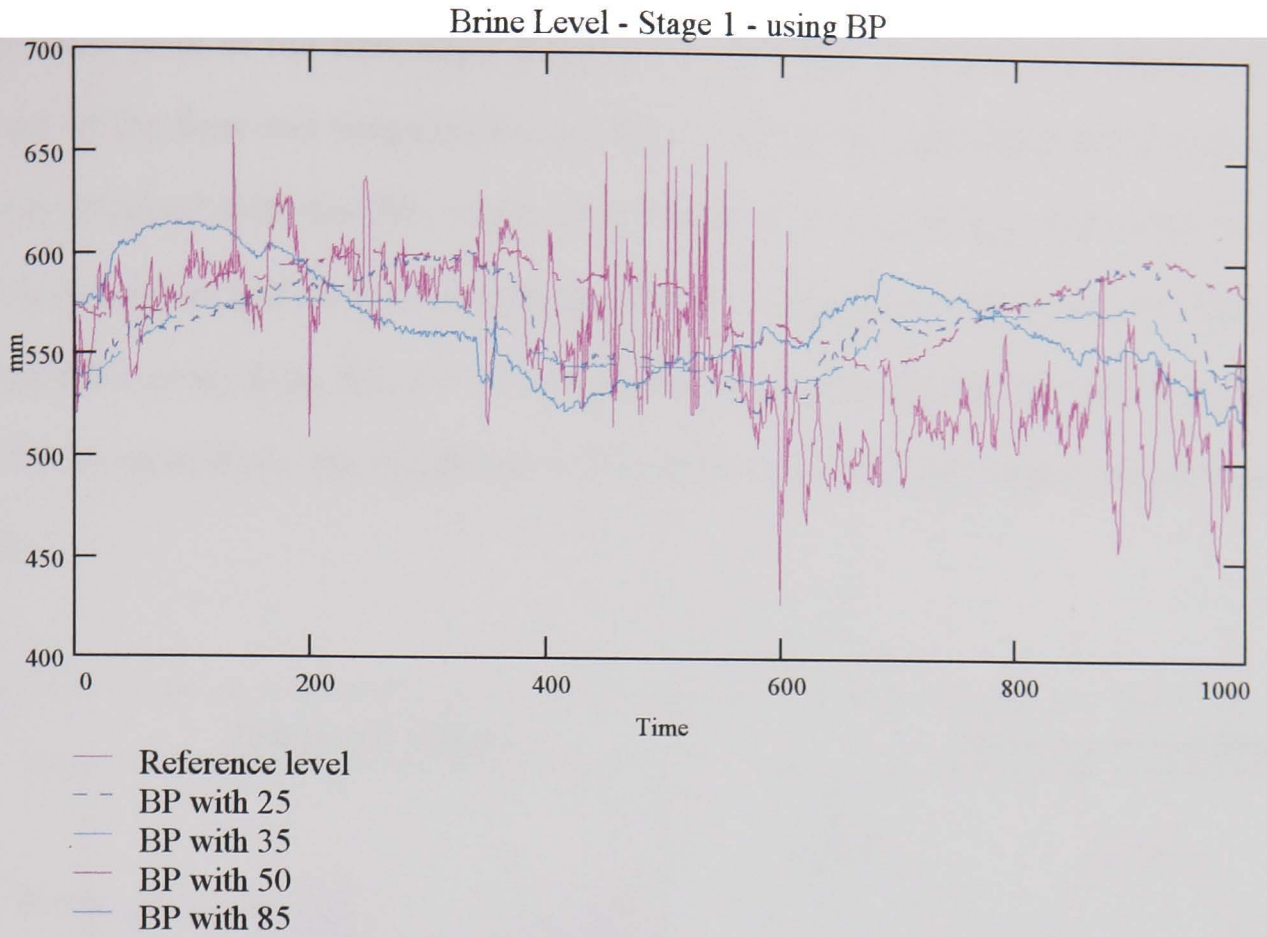


Figure 5.7: A comparison between BP learning algorithm with different hidden neurons number for prediction of the first stage level.

5.5.2 GENE Approach

The BP learning algorithm is shown to give good approximation, with the exception to prediction of the first stage brine level. When using the GENE approach for training, it is found that a minimum number of hidden neurons is required otherwise the network will never learn. The minimum hidden neurons number is estimated by directly performing the learning phase with low number of neurons and if the error is not reduced, then add more neurons, until one reaches the number at which the error is reduced suddenly. Test results for using 60 nodes for each of the two hidden layers are shown in figures 5.8 & 5.9. The prediction of the first stage brine level and the last stage distillate level are shown to be in agreement with the data obtained as compared to the prediction obtained with the BP learning algorithm. Moreover, the estimated level follows the oscillatory characteristic of

the brine level in the first stage obtained in the data. Furthermore, figure 5.9 shows the result of the flow and temperature variables with an increased performance as compared to those obtained with the BP. Notice that for both cases, the last stage brine level is shown to be smoother and do not have the oscillatory characteristic because the brine blow down is used for controlling the level in a closed loop. By increasing the number of hidden nodes to 75 for each layer, the prediction of the brine level is further improved as shown in figure 5.11.

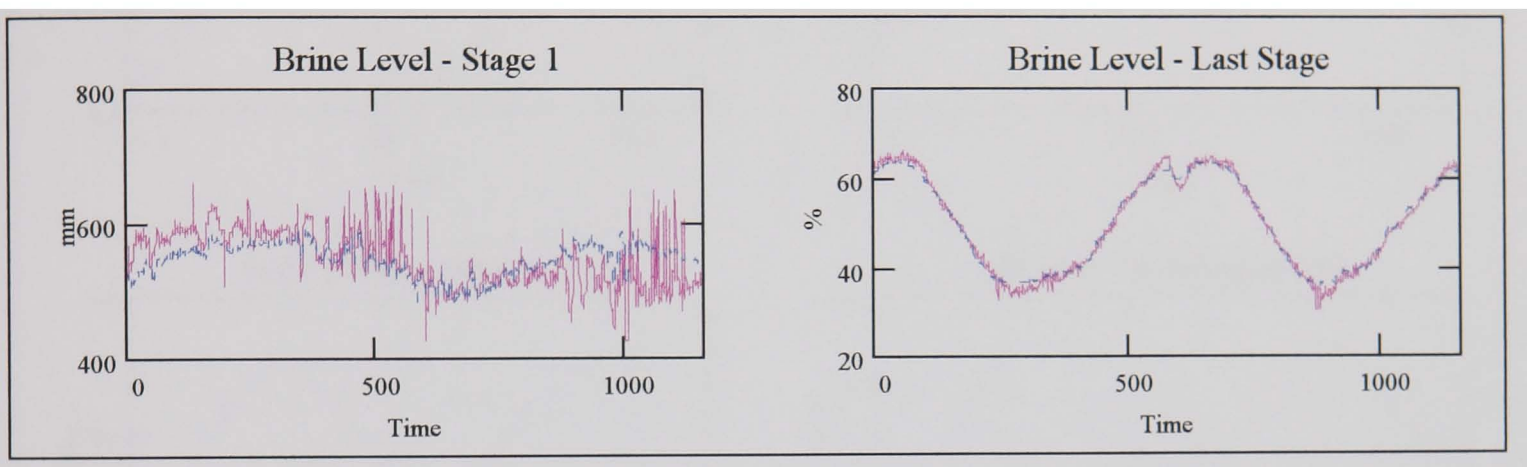


Figure 5.8: Test results for modelling MSF plant with 16 stages using GENE learning algorithm with 60 neurons for each hidden layer.

The dotted lines represent the network output and the lines represent results from the actual plant.

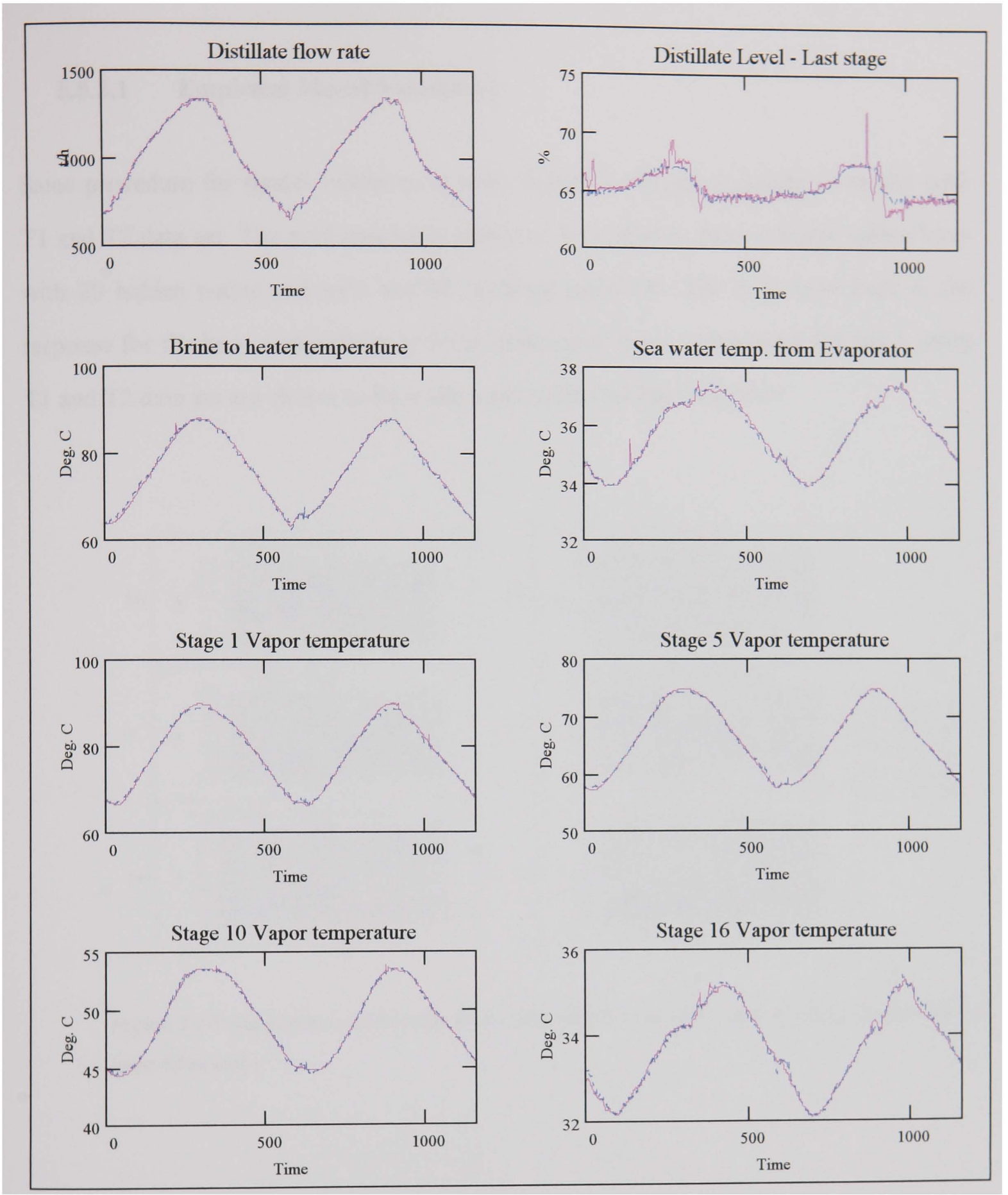


Figure 5.9: Test results for modelling MSF plant with 16 stages using GENE learning algorithm with 60 neurons for each hidden layer.

The dotted lines represent the network output and the lines represent results from the actual plant.

5.5.2.1 Empirical Model Validation

Same procedure for model validation is used. Figure 5.10 shows simulation results with T1 and T2 data set. The performance is shown to be similar to have a single hidden layer with 20 hidden nodes and with the BP learning algorithm. The only draw back is the response for the brine temperature to brine heater and vapor temperature for cell 1 using T1 and T2 data set are shown to have the same values during simulation.

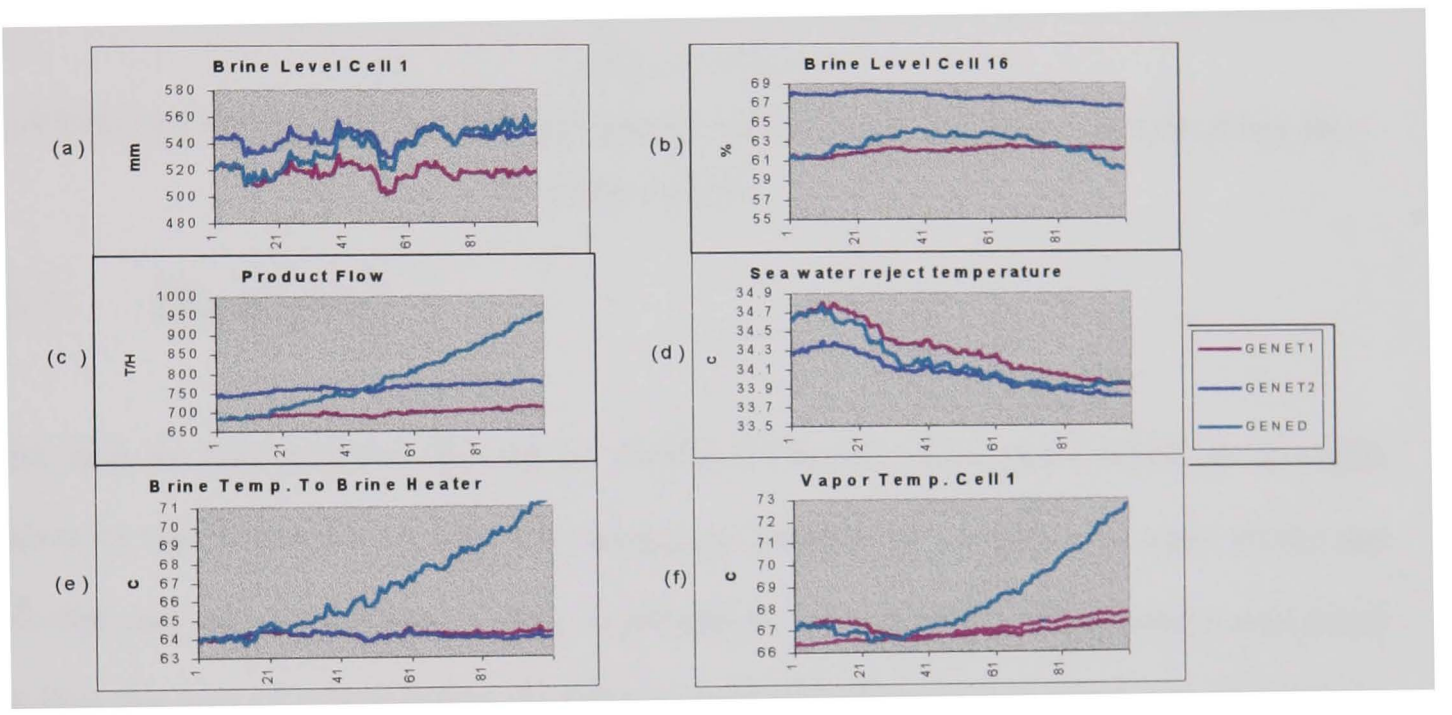


Figure 5.10: Simulation results from ANN using GENE approach, each of the two hidden layers have 60 neurons

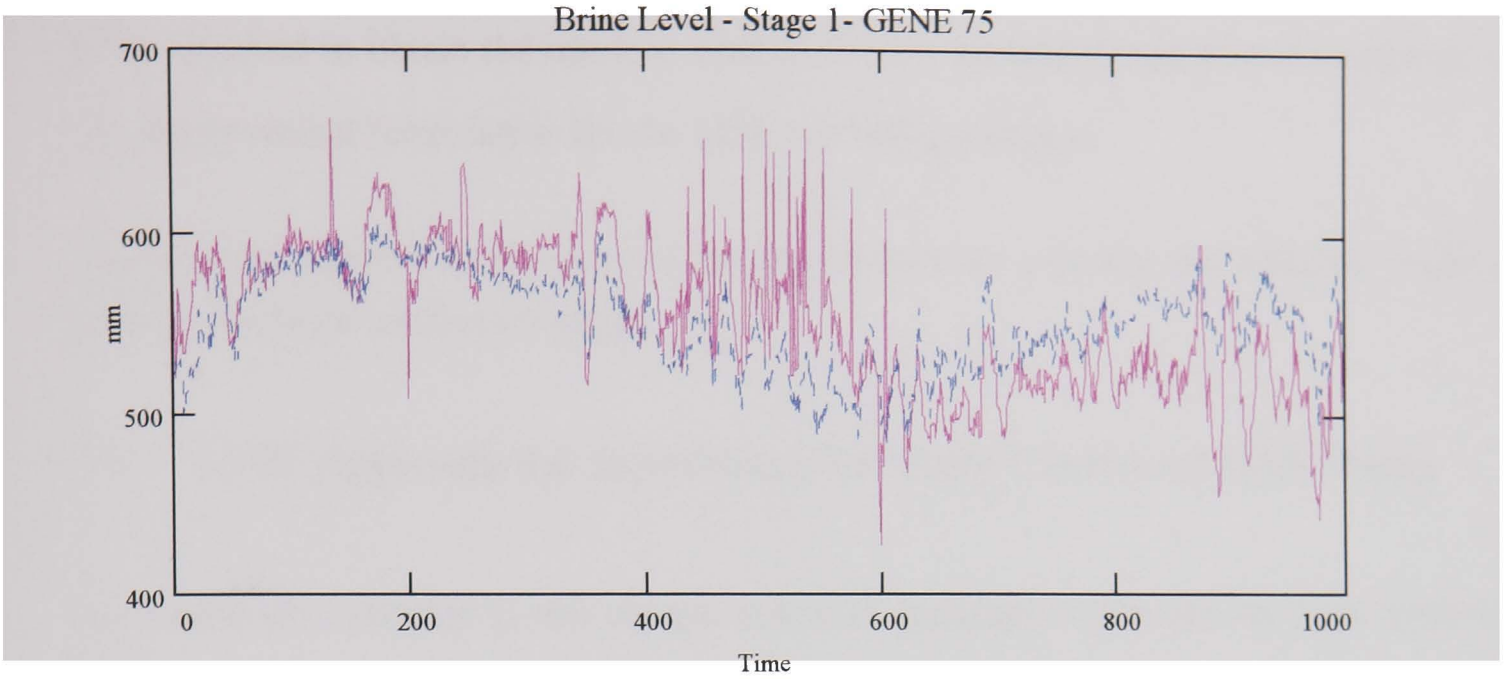


Figure 5.11: Prediction of the first stage brine level using GENE approach (2 hidden layers with 75 nodes each)

The dotted lines represent the network output and the lines represent results from the actual plant.

5.5.3 Discussion

With the aim to demonstrate the use of Artificial Neural Networks (ANN) as a viable alternative to mathematical models for numerical simulation purposes, a case study has been carried out on a multistage flash desalination plant (MSF). The results described suggest that the use of ANN based on GENE approach for training, rather than with the standard BP learning algorithm, is adequate for modelling the MSF process and prediction of the stage brine level during load variation over the entire range of data obtained. If the brine levels were not required for the model, then the BP algorithm with single hidden layer would be enough for such a problem. Another approach is to use the BP for modelling the process variables, but for the brine levels, to use the GENE approach. While the plant considered is equipped with brine level measurements for stage 1 and stage 16 only, if a complete model covering all the brine levels for all stages is required, then level measurement for all brine level stages would be necessary. It is worth to note the small

effort required to obtain the result as well as the low computational power compared with any mathematical formulation for the MSF modelling problem.

Choosing the correct architecture is the key to have an accurate and efficient model that predicts the brine level in all stages.

5.6 ANN Approach for Supervisory Set Point Control of MSF Plants

As discussed in chapter 2, the current practical applications for the set point generation program is based on availability of an approximate operating curves for the brine recirculation flow as a function of the product distillate flow rate and the cooling sea water temperature. Additionally the TBT is calculated using the heat and mass balance equations for the evaporator in their linear form. The operating curves are usually obtained during the first commissioning of the units, and may require further tuning to either enhance the performance or due to slow changes in the plant parameters that may occur with time. However, parameter tuning and updating is usually based on trial and error. The tuning process requires the process engineer to be familiar with all programming and tuning details and special skilled people are required to be involved. However, an engineer who understands the process being simulated requires a few days training in the neural network approach. For the purpose of plant modelling with the ANN approach to learn the data from the plant is required.

This section is devoted to demonstrate the applicability of ANN approach to learn and generate the necessary set point steps required for load variation between 60% and 100% load production for MSF plants. The developed procedure includes a method to validate the set point generation based on the capability of ANN to predict a practical comprehensive range of the process behavior.

5.6.1 Procedure for Set Point Generation using ANN Approach

To learn the set point generation, a trajectory is developed to cover comprehensive operating points during load variations. This is implemented by considering the sample of the system input-output during load variations which follow a time state space function V(t) with a mapping function NNE that can be estimated or approximated using ANN approach. Three networks are used for set point control generation and validation. The procedure is based on cascaded architecture of ANN and they are used as an Estimator network (NNE), Controller network (NNC) and a Mapper network (NNM). The block diagram of the approach is shown in figure 5.12. The procedure can be described in the following:

1. The nonlinear time state space trajectory is estimated for different operating points (input and output data) during load variations using NNE. The performance of the network is studied by considering the time state space trajectories V(t) which is composed of several vectors, and comparing the performance to the various combinations of vectors.

2. The control action is learnt using this nonlinear time state space trajectory with NNC. NNC could be one network or sub-divided in some cases into two sub-networks. In this case, there are two sub-networks. The first sub-network (NNCI) is used for control during load increase and the second (NNCD) is used for control during load decrease. The performance of the network is studied to find the case when only one network of NNC is used to learn the set points required to increase and decrease the load simultaneously.

3. To validate the final results, NNC is used to generate the control signal to the NNM network based on the estimation of the first network NNE and to compare NNM behavior to the process data.

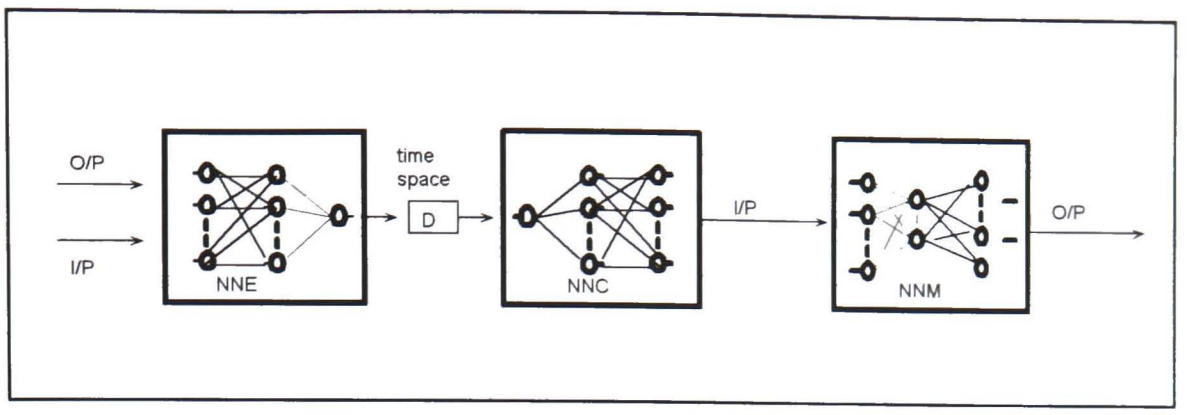


Figure 5.12: The block diagram based on cascaded architecture of ANNs and they are used as an Estimator network (NNE), Controller network (NNC) and a Mapper network (NNM).

5.6.2 Data Collection, Preparation and Preprocessing (Scaling)

For plant modelling and set point generation, data was obtained for 60%-100% load variation of a MSF plant (13 recovery stages and 3 reject stages) for every 30 seconds for the following variables:

1. Brine recirculation flow rate;
2. Top brine temperature (outlet from the brine heater);
3. Makeup seawater flow rate;
4. Steam input flow;
5. Seawater circulation rate to maintain the sea water inlet temperature during winter conditions;
6. Sea water temperature;
7. Brine blow down flow rate;
8. Steam input temperature;
9. Brine level in the last stage;
10. Distillate product;
11. Brine temperature to brine heater; and
12. Seawater from reject section temperature.

Variables 1 to 7 are considered as input to the model, and the other five are the corresponding output variables. Data covering load increase and decrease are included

with the vapor temperatures in each stage, so that more sensory data are used for the training. LP steam temperature is 182.5 DEG C and with a pressure of 0.75 bar. The maximum and minimum values used during the training are shown in Table 5-4.

5.6.2.1 Time State Space Trajectory for Set Point Variation Control

The *time state space* trajectory for set point variation control contains several vectors {v1, v2, v3,...vn}, where each vector is introduced to either define the status of the plant to the maximum / minimum load, or to define the plant logical condition like the load variation type (increase/decrease). To define the status of the plant to the maximum / minimum load data points, each collected data point is tagged with a step number (in the data set during load variation). This means tagging each collected sample or pattern with a number (x) which is successively increased / decreased with the next collected sample, depending on the load changing condition. The status of the plant can now be defined from the collected data, and by choosing a suitable variable proportionally related to the load (the top brine temperature (TBT) in our case), the maximum and minimum normalized values are selected as x_{max} and x_{min} respectively. They are considered as the upper and lower normalized steady state load status values. Now the difference q between any two successive (x) is calculated as follows:

$$q = (x_{max} - x_{min}) / k \qquad (1)$$

Where k is an integer and represent the total number of data collected during load variations.

Therefore, the following vectors with dimension kx1 can be used to define the time state trajectory:

$$v1 = [x_{max}, x_{max-q}, \ldots\ldots x_{min}]^T \qquad (2)$$

$$v2 = [0,\ldots q,q,\ldots,0,\ldots -q,-q,\ldots\ldots 0]^T \qquad (3)$$

$$v3 = [\alpha_{max},\alpha_{max-q},\ldots\ldots,\alpha_{min}]^T \qquad (4)$$

$$\nu 4 = [\sigma_1, \sigma_2, \ldots\ldots\ldots\ldots\ldots\ldots \sigma_k]^T \qquad (5)$$

Where

$$\alpha = \tanh(x) \qquad (6)$$

σ is the average of seawater inlet, make-up, blow down, and seawater recirculation flow variables.

The time state can now be defined according to the values of $\nu 1$, $\nu 2$, $\nu 3$ and $\nu 4$, i.e. in the space represented by these vectors. If two vectors represent the time state, then it is a two-dimensional state and is called time state plane. When the representation uses more than two vectors, it is called the time state space. For simplicity, the time state space terminology will be used in the text.

Where

$\nu 1$ represents a linear relationship between the time and the status of the system.

$\nu 2$ represents the condition of load (increasing, decreasing or no change).

$\nu 3$ represents a nonlinear relationship between the time and the status of the system and takes the shape of the tanh function between the minimum and maximum load.

$\nu 4$ represents a variable parameter from the plant.

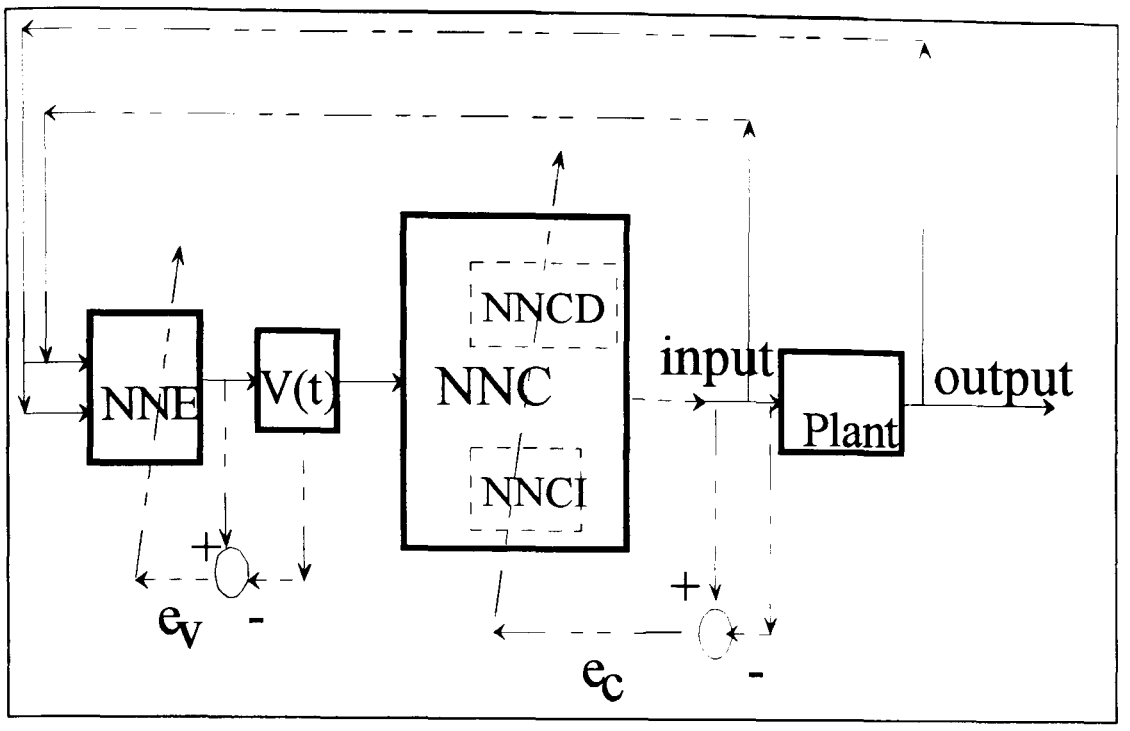


Figure 5.13: Block diagram of the proposed neural network for step estimation and control of MSFD plant

5.6.3 Training Consideration

In this study all networks used have single hidden layer. It is found that by using the back propagation learning algorithm satisfactory required performance has been achieved. The data used for training does not include the brine level in the first stage.

As shown in figures 5.13 and 5.14, the block diagram of the proposed NN controller consists of two networks: an NN Estimator (NNE), and an NN controller (NNC). NNC is sub-divided into two sub-networks, (NNCI) for load increase and (NNCD) for load decrease. By applying both input and output data to the NNE, the output of this network is the time space state, which is the subsequent input to NNC. Now the output of NNC contains the values of the input data to the plant (set point to regulatory control), is applied to NNM to produce the desired output of the plant.

During the training phase, each network is independent of each other. While in the application phase, by increasing / decreasing the time space states successively with a

delay (D) will produce the required trajectory of the output load with respect to the required control input. Since the delay is considered as the sampling rate of the obtained data, hence the dynamics of the system are included. The set point control is generated from both input-output data of the plant. In this case D = 30 seconds. The weights and threshold were initialized randomly to lie within (± 0.01). The NNC was a 4-20-7 network while NNE was 28-20-4 and NNM was 7-20-5 with the hyperbolic tanh function in the hidden and output units.

Considering the sample of the system input - output during load change following a time state space function V(t) with a mapping function N that can be approximated using NN, the following cases are considered:

Case 1: $V(t) = N[y(t),u(t)] = \{v1,v2\}$ (7)

Case 2: $V(t) = N[y(t),u(t)] = \{v1,v3\}$ (8)

Case 3: $V(t) = N[y(t),u(t)] = \{v2,v3\}$ (9)

Case 4: $V(t) = N[y(t),u(t)] = \{v1,v2,v3\}$ (10)

Case 5: $V(t) = N[y(t),u(t)] = \{v1,v2,v3,v4\}$ (11)

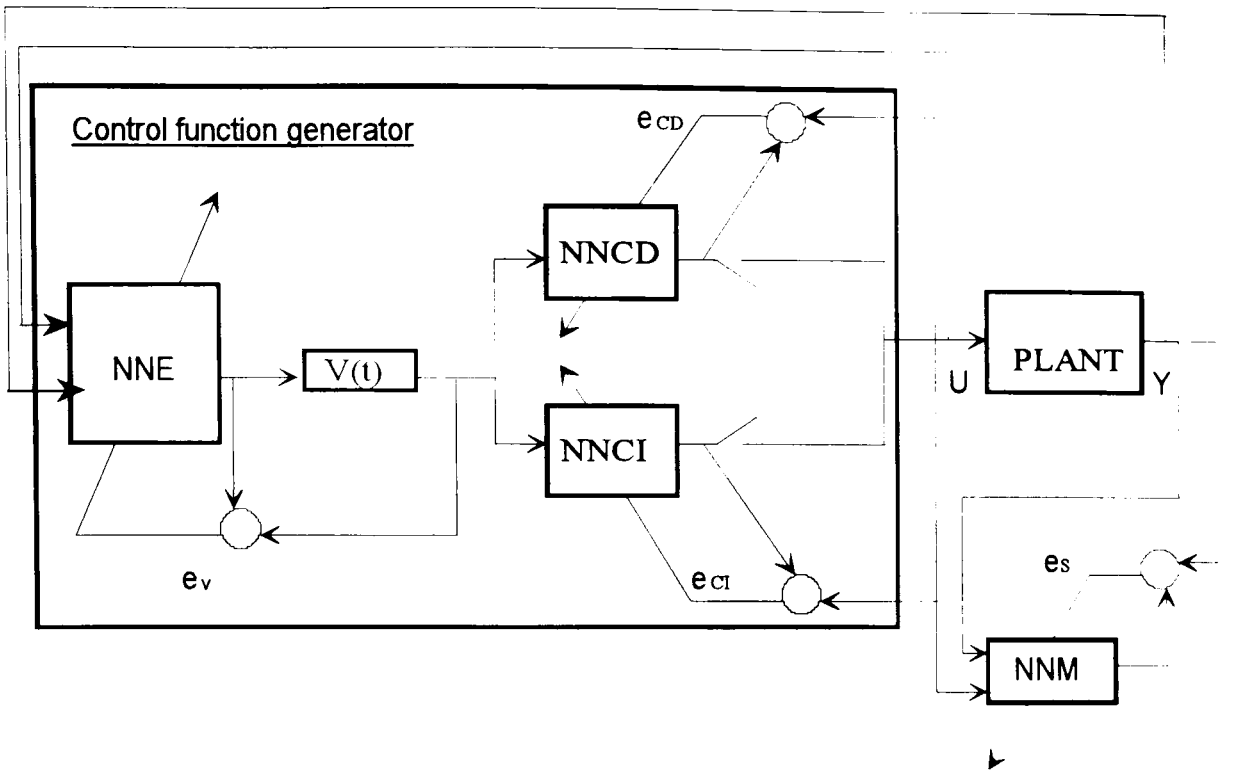


Figure 5.14: Block diagram of neural network with back-propagation training algorithm for modelling and set point control function generation for load variation of MSF unit

The collected input and output measured variables are applied to NNE where its estimated output is compared to the time state space function V(t). So NNE can be employed by using the error between the network output and the nonlinear time state space trajectory denoted by e_V as shown in figure 5.13.

The performance index

$$J_E = \tfrac{1}{2}\sum_i r_i (y_i(t) - \hat{y}_i(t))^2 \qquad (12)$$

is used in the training where $\hat{y}_i$ is the *i*th output of the sub-network.. The weights in NNE are updated as follows:

$$W_E(t)=W_E(t-1)-\Xi_E(W_E) \qquad (13)$$

where the subscript E refers to NNE, and $W_E(t)$ is the weight vector consisting of all weights in the sub-network at time t. The quantity $\Xi_E(W_E)$ is the gradient vector, consisting of the derivatives of J_E with respect to each weight in the NNE, and r_i is the step length for weight updating.

5.6.4 Control Generation Simulation

Increasing and/or decreasing the load of a MSF desalination unit requires generation of pre-determined steps to the various set points of the regulated variables. The final required set points (steady state condition) is calculated by the mathematical model, while choosing the most applicable steps is by trial and error [18, 19]. To generate the intermediate set points, ANN is proposed to approximate the set-points step generation based on input/output samples collected during load change. Firstly, and after sufficient learning by introducing all input-output data to (NNE), the output result of NNE, V(t), is used as input to (NNC).

By applying V(t) to the NNC network, the output of this network is the control input to the plant during load increase as well as load decrease. The step length, momentum, and temperature coefficients are chosen as shown in table 5.5. Results of variation of these coefficients for NNM network are shown in figure 5.15. Results on Figures 5.16, 5.17 and 5.18 show that one NNC controller could not approximate the control function by using V(t). The RMS error could not be reduced to an acceptable level.

Considering cases 1,2,3 and 4, the learning ability appears only after splitting the NNC controller into two parts: NNCI and NNCD controllers (see figure 5.14) are for load increase and load decrease respectively. Figures 5.19 and 5.20 show that the RMS results are satisfactory (RMS < 0.1). Variations of the learning rule are considered in all cases (Delta rule, Normalized delta rule and Commulative delta rule). The comparison shows that by using the commutative delta rule for the back propagation algorithm has not improved the network performance in reducing the error function. Whilst by using either delta rule or the normalized delta rule the performance of the network has improved by reducing the error function.

Now using V(t) as defined in case 5 to avoid splitting of the controller, the estimated results and desired values are shown in figures 5.21 and 5.22. It indicates that NNC is capable of reducing the error between the network output and the plant input denoted by e_c in figure 5.13 and with similar performance index and weights update as shown in equations (12) and (13).

5.6.5 Discussion

A case study and simulation results using real data from the plant have demonstrated the capability of the proposed neurocontroller when it is applied to the process under consideration. Comparative simulation results have shown that by adding more sensory information with proper choice of the learning rate and other learning parameters, the neurocontroller could generalize all cases available. The above results are based on a new approach of using the neural networks for control of MSF desalination plant.

These results are for variations from 60% to 100% load a satisfactory mapping for the input-output relationship has been achieved. The network could be extended to learn more data when necessary. The foregoing shows that the most common task of the ANN is to perform a mapping from an input space to an output space and can approximate the nonlinear function between the input-output relation accurately. The approach provides the procedure to implement NN controller to satisfy the requirements of MSF tracking problem based on cascaded architecture of NN. The NN controller consists of two networks, which are used for time-state space trajectory estimation and control signal generator, respectively. The superior performance of ANN is not limited to model the process behavior (NNM), but also to learn the generation of the control action from the data obtained.

5.7 Conclusions

The use of ANN based on the GENE approach for training has been found to give good results in prediction of the MSF stage brine levels.

Investigation have shown that, under some conditions, and using all the available information with the GENE approach adopted for the MFN, the network will give accurate results with a minimum effort for the data obtained during load change (transient condition). It is evident that this is true for data obtained for such dynamic behavior of MSF plant during load change, particularly for the brine level. If the brine levels were not required for the model, then the BP algorithm with single hidden layer would be enough for such a problem. Another approach is to use the BP for modelling the process variables, and for the brine levels to use the GENE approach.

Compared with other reported work [67], we have utilized real data obtained from the plant so that simulation and test result gives an actual behavior of the plant.

In the next chapter, disturbances that cause different instability modes for MSF plant and all major loops will be explored. Separate dynamic tests were conducted on existing plant to obtain system dynamic behavior for each control loop. This causes instability and low performance of the trained network and the solution to this will be explored in the next chapter.

VARIABLES	Min. value	Max. value	units
Brine Recirculation flow	13210	15000	t/h
Brine from heater temperature (TBT)	67.78	94.73	deg C
Sea water (SW) inlet flow	5003	7229	t/h
SW to evaporator temperature	27.81	32.75	deg C
SW circulation flow	5667	8438	t/h
Makeup SW flow	3266	6090	t/h
Brine blow down flow	2261	5125	t/h
Steam input flow rate	77.42	171.9	deg C
Brine Level in first stage	426.3	662.7	mm
Brine Level in last stage	30.3	66	%
Distillate product flow	648.6	1350	t/h
Brine to heater temperature	62.59	71.98	deg C
Distillate Level in last stage	63.02	87.92	%
SW from reject section temperature	33.85	37.51	deg C
Vapor temperature in stage 1	66.24	89.95	deg C
Vapor temperature in stage 2	64.58	86.74	deg C
Vapor temperature in stage 3	62.57	82.89	deg C
Vapor temperature in stage 4	59.82	78.77	deg C
Vapor temperature in stage 5	57.26	74.84	deg C
Vapor temperature in stage 6	55.15	70.81	deg C
Vapor temperature in stage 7	51.09	66.23	deg C
Vapor temperature in stage 8	49.66	62.11	deg C
Vapor temperature in stage 9	46.45	57.81	deg C
Vapor temperature in stage 10	44.26	53.78	deg C
Vapor temperature in stage 11	42.06	49.93	deg C
Vapor temperature in stage 12	40.5	46.64	deg C
Vapor temperature in stage 13	38.21	42.79	deg C
Vapor temperature in stage 14	36.34	40.14	deg C
Vapor temperature in stage 15	34.28	39.48	deg C
Vapor temperature in stage 16	32.13	35.32	deg C

Table 5.1: Variables minimum & maximum values

	Output Layer	Hidden Layer
Step length	0.15	0.3
Momentum	0.4	0.4
Temperature	0.1	0.1

Table 5.2: Learning schedule for output & hidden layers

variable	Brec	TBT	MAKEUP	STF
Dimension	t/h	^{o}C	t/h	t/h
min.	13250	68	3860	89
max.	15000	80	5240	149

Table 5.3: Minimum & maximum values adopted for the input variables during model validation simulation tests T1 and T2

VARIABLES	Min. value	Max. value
Sea water make-up flow (t/h)	3860	5906
Cell 16 distillate level (%)	65.88	67.53
Brine Recirculation flow (t/h)	13254	14726
Low pressure steam flow (t/h)	89.32	162.9
Sea water to evaporator flow (t/h)	5362	7079
Sea water recirculation flow (t/h)	7418	7491
Brine blow down flow (t/h)	3437	4594
Brine from heater temperature (deg C)	68.8	94.66
Brine to heater temperature (deg. C)	63.9	87.92
Sea water to evaporator temperature (deg C)	30.34	31
Sea water from evaporator temperature (deg C)	34.55	36.71
distillate flow (t/h)	701.7	1333

Table 5.4: Variables minimum & maximum values

	Output Layer				Hidden Layer			
Learn count	6000	12000	19000	25000	6000	12000	19000	25000
Step length	0.15	0.4	0.019	0.001	0.3	0.15	0.038	0.002
Momentum	0.4	0.2	0.05	0.003	0.4	0.2	0.05	0.003
Temperature	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1

Table 5.5: Learning schedule for output & hidden layers

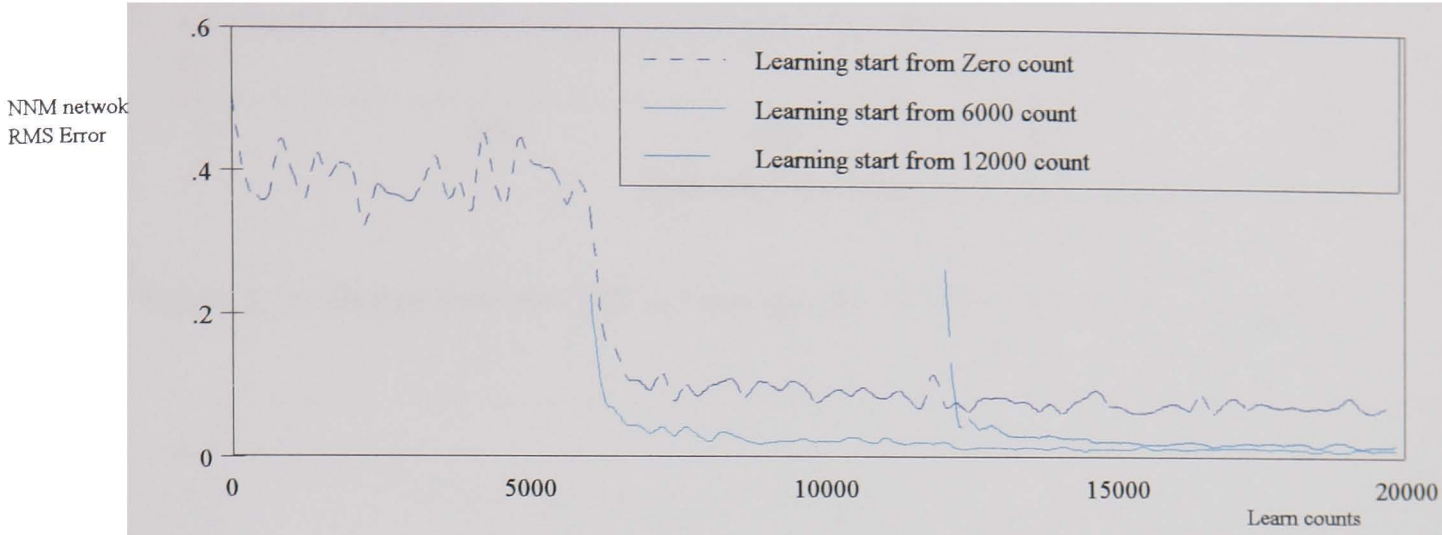


Figure 5.15: Comparison of output error for NNM net during training with different learning rate at the starting time. Learning is automatically changed with the learn counts as per table 2.

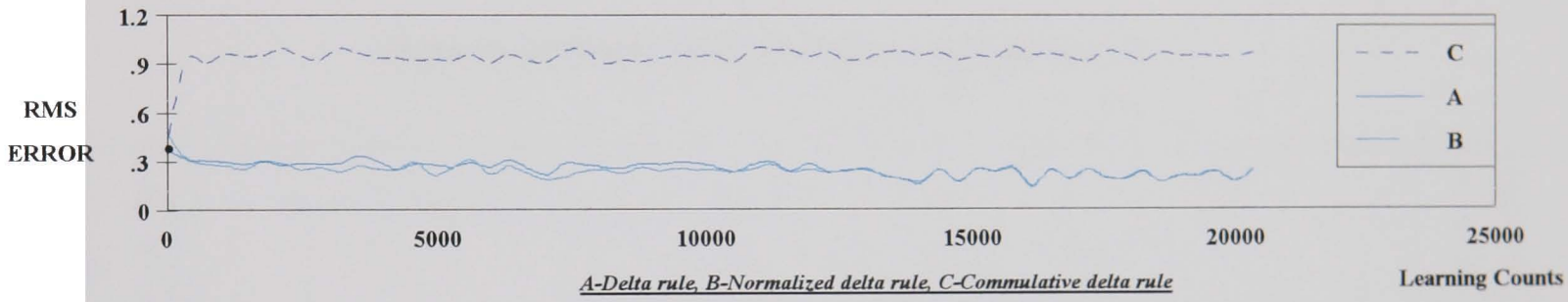


Figure 5.16: Output error for NNC net with epoch = 35, with V(t)={V1,V2} as the input

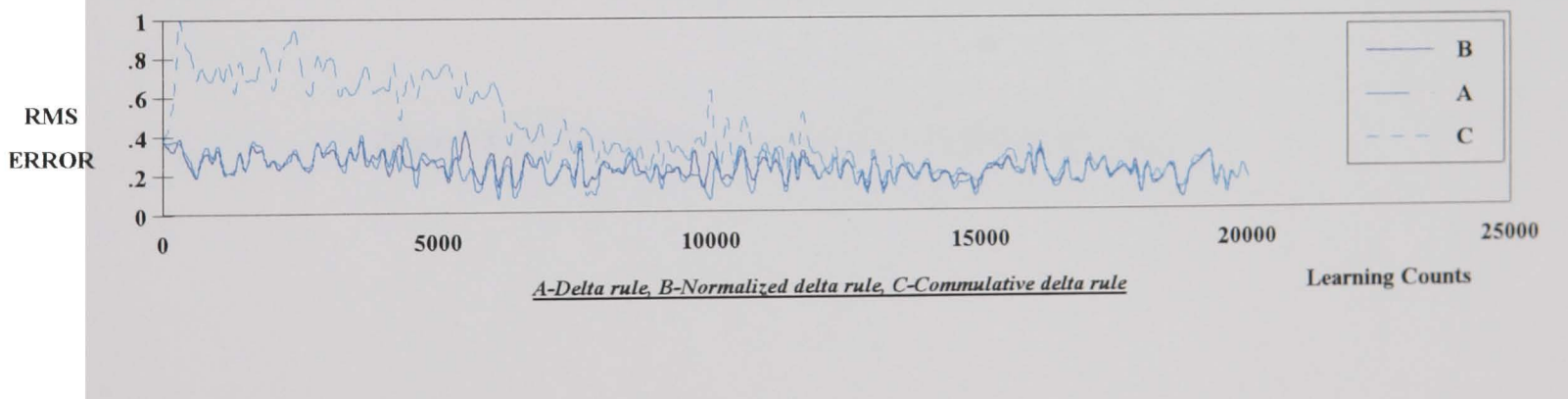


Figure 5.17: Output error for NNC net with epoch = 10, with V(t)={V1,V2} as the input

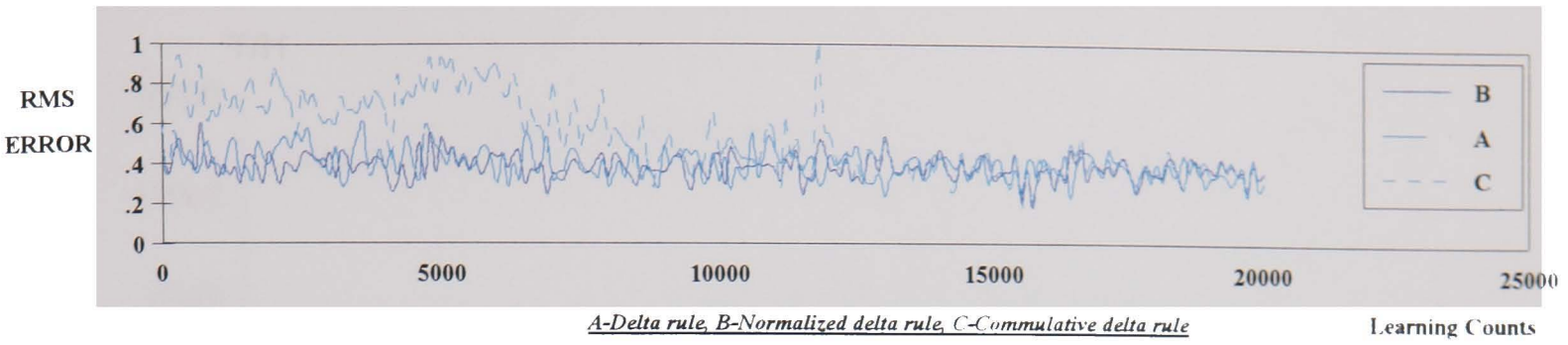


Figure 5.18: Output error for NNC net with epoch = 10, with V(t)={V1} as the input

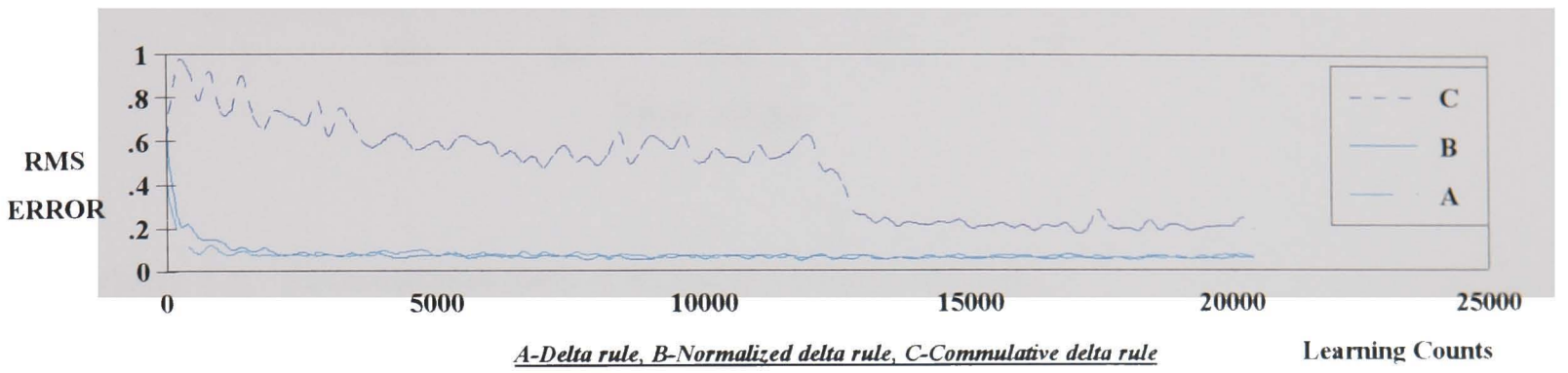


Figure 5.19: Output error for NNCI net with epoch = 20

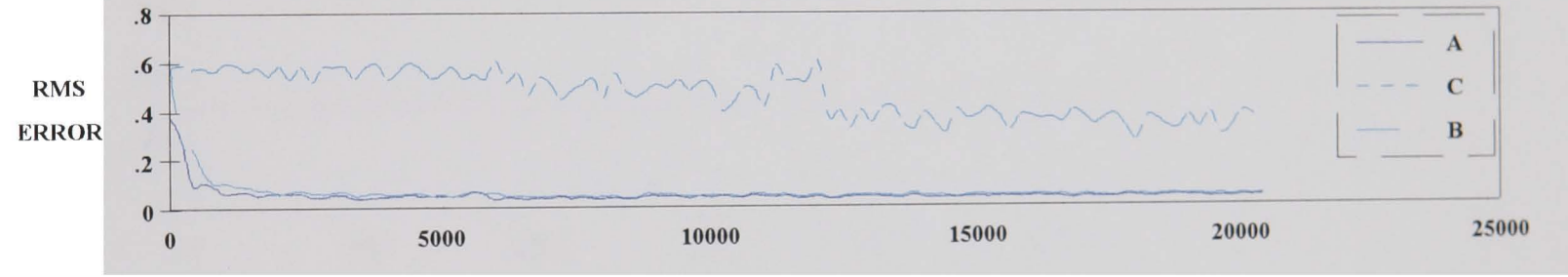


Figure 5.20: Output error for NNCD net with epoch = 20

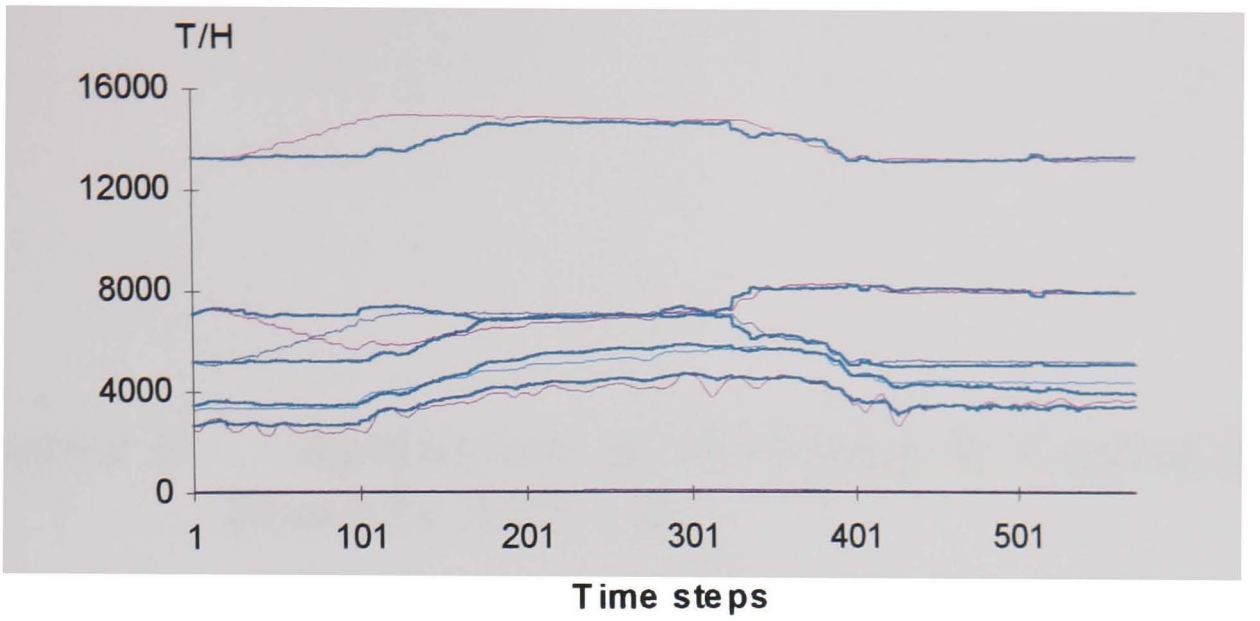


Figure 5.21: Estimated (dark lines) & desired (doted lines) flow values

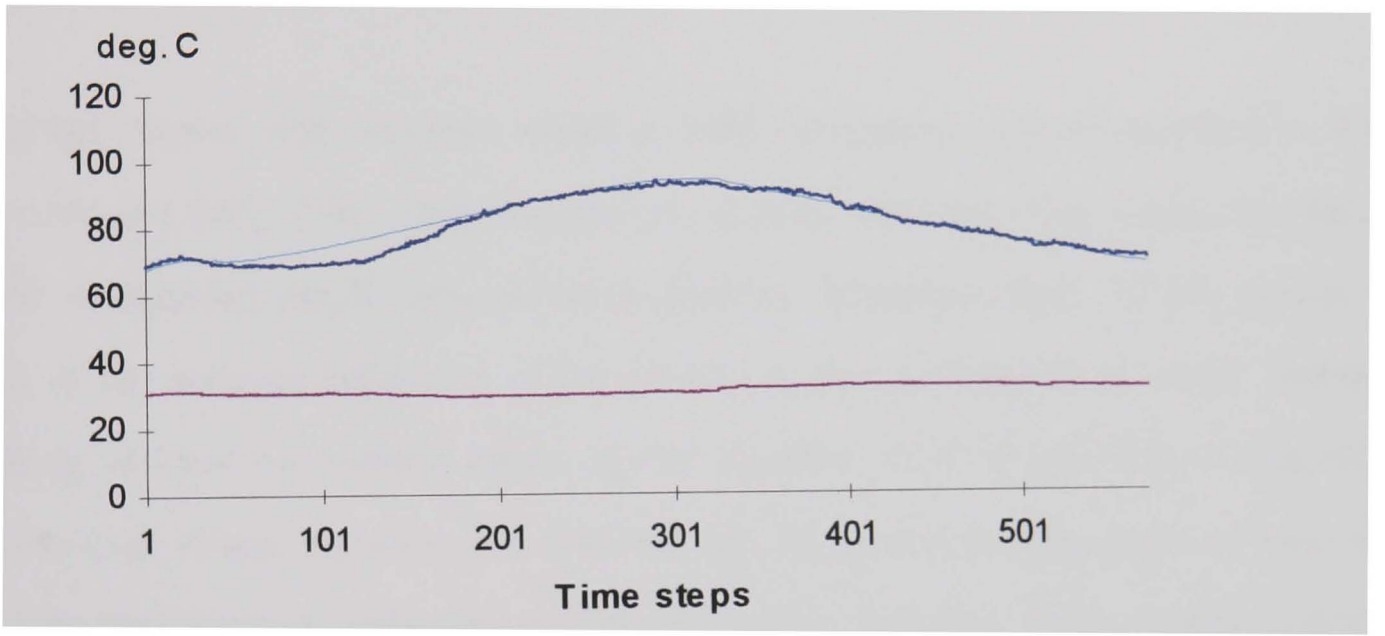


Figure 5.22: Estimated (dark lines) & desired (doted lines) temperature values

Chapter 6 Application of Modelling & Control of MSF Plants by ANN - II

6.1 Introduction

In the last chapter, other reported works on MSF desalination process modelling by ANN was reviewed [67], [68]. They focused on systems obtained from simulators, though results considering simple network were reported. Moreover they did not discuss the effect of the dynamic behaviour of the system on the performance of ANN. Generally speaking, at least two control modes can be identified for MSF; the first is required to perform load change between minimum and maximum load working point to cope with the daily load demand, while the second mode is the response of the control system to external disturbances such that the plant availability is not disturbed. The first one has been dealt with in the last chapter and difficulty is experienced in capturing accurate prediction of the first stage brine level. To investigate the second mode for MSF desalination plant, all main control loops were observed during imposed disturbances so that the interactions of the desalination process could be covered. The objective of this chapter is to present a dynamic black box model of the desalination process and to provide a method for learning the control action due to external disturbances.

In this chapter we present a different case from the prevoius one, namely the application of MFN to MSF desalination plant to identify and model the process behaviour due to dynamic disturbances. Specifically our attention will be focused to cover the major control loops, and they are as follows:

- Top brine temperature control loop.
- Brine recirculation flow control loop.
- LP steam temperature control loop.
- Brine level control loop.
- Distillate level control loop.
- Sea water make up flow control loop.
- Sea water supply temperature control loop.

To obtain the relevant information about the process behaviour, dynamic tests were necessary. The unit steps were made at the control valve, e.g. to realize a sudden change of the steam flow to the brine heater, then leave the system to stabilize by itself due to such disturbance. Thus the dynamic test involves disturbing or exciting the manipulated variables for some control loops. By such excitation the resultant effect on the controlled variables could be obtained. Seven different dynamic scenarios were possible to be obtained as follows:

1. Brine recirculation dynamic test (dt_1)
2. Brine level dynamic test (dt_2)
3. Distillate level dynamic test (dt_3)
4. Low pressure steam flow dynamic test (dt_4)
5. Low pressure steam temperature dynamic test (dt_5)
6. Makeup flow dynamic test (dt_6)
7. Sea water supply temperature dynamic test (dt_7)

Additionally a static test (dt_8) (slow variation of the process variables such that no disturbances are encountered) could be obtained for the following variables:

- Brine recirculation flow.
- Brine level in last stage.
- Distillate level in last stage.
- Sea water supply temperature.

The foregoing scenarios will be used for network training and to study the network model performance. The philosophy of applying ANN involves training on part of the scenarios obtained, and then, using the rest of the scenarios to test and study the network performance. The GENE approach is adopted as the learning algorithm, for which it is required to investigate what dynamic capability could be provided by the feed forward network. The internal behaviour and convergence properties of the algorithm are compared to the standard back propagation with one hidden layer, and the advantages and disadvantages are discussed. The results are illustrated using data obtained from Al-Taweelah MSF plant at Abu Dhabi. The objective is to perform the learning using limited number of data and to generalize for every excitation scheme obtained, so that the dynamics of the process is learnt efficiently. Moreover, identifying which of control loops are critical to learn the plant dynamic due to external disturbances.

6.2 Dynamic Test and Data Collection

Generally speaking, process disturbances will cause one of the following effects:

1. The process is disturbed such that the internal process is excited. The excitation effect is reduced and drives the process to the same or another stable state.
2. The process excitation continues cyclically or exponentially reaching unstable state.

For plant stability, the second effect is not desirable. All tests conducted on the process to obtain the data fell under the first effect. The system was excited by 5 to 10% of the operating point (a unit step of variable magnitude and duration time is used). Examples were generated from Al-Taweelah MSF plant at Abu Dhabi. Table 6.1 (a,b) contains list of data points obtained for the seven control loops and additional monitoring points.

1. Top Brine Temperature Controller	Range
1. Top brine temperature set point	0-250^{o} deg C
2. LP steam flow	0-250^{o} deg C
3. Top brine temperature controller output	0-100 %
4. LP steam valve position	0-100 %
2. Brine Level Controller	
5. Brine level controller set point	0-1000 mm
6. Brine level last stage	0-1000 mm
7. Brine level controller output	0-100 %
8. Blow down valve position	0-100 %
9. Brine level controller blow down flow	0-8000 tons/h
3. Distillate Level Controller	
10. Distillate level controller set-point	0-1000 mm
11. Distillate level	0-1000 mm
12. Distillate level controller output	0-100 %
13. Distillate level valve position	0-100 %
4. Sea Water Make Up Flow Controller	
14. Seawater make up flow controller - set ratio	0-8000 tons/h
15. Seawater make up flow	0-8000 tons/h
16. Distillate flow	50-2000 tons/h
17. Seawater make up flow controller output	0-100 %
18. Seawater make up valve position	0-100 %
5. Seawater Supply Temperature Controller	
19. Seawater supply temperature controller set-point	0-20000 tons/h
20. Cooling seawater flow	0-20000 tons/h
21. Seawater supply temperature controller output	0-100 %
22. Cooling seawater valve position	0-100 %
6. Brine Recirculation Flow Controller	
23. Brine recirculation flow controller - set Ratio	0-20000 tons/h
24. Brine recirculation flow	0-20000 tons/h
25. Brine recirculation flow controller output	0-100 %
26. Brine recirculation valve position	0-100 %
27. Brine heater outlet valve position	0-100 %
7. LP Steam Temperature Controller	
28. LP steam temperature controller set Point	0-150^{o} deg C
29. LP steam temperature controller output	0-100 %
30. LP steam temperature after spraying	0-150^{o} deg C
31. Seawater valve Position	0-100 %

Table 6.1(a): List of the data points for the seven control loops obtained during the various dynamic tests

Additional Monitoring Points	Range
32. Brine heater inlet temperature	0-120^{o} deg C
33. Brine heater condensate level	0-1000 mm
34. Brine recirculation inlet temperature	0-120^{o} deg C
35. M.P. steam temperature	0-250^{o} deg C
36. Spray water flow	0-15 tons/h
37. Cooling seawater temperature	0-50^{o} deg C
38. Seawater temperature	0-50^{o} deg C
39. Condensate temperature	0-120^{o} deg C
40. Brine level of first stage	0-1000 mm
41. Seawater recirculating flow	0-4000 tons/h
42. Seawater make up temperature to deaerator	0-60^{o} deg C
43. Top brine temperature	0-120^{o} deg C
44. LP steam temperature	0-150^{o} deg C

Table 6.1(b): List of the additional monitoring points obtained during the various dynamic tests

Due to the high amount of measurement signals and long duration period of every test (every change of the top brine temperature needed approximately 30 to 45 minutes) the measurement system had to deal with lots of data. The hardware used includes data acquisition system featuring 126 measuring channels (16-bit resolution) with a sampling rate of 10 kHz for 16 channels could be connected via measurement racks to a controlling and storing computer. For rapid functions 16 channels are available (12 bit resolution) with a scanning frequency of 1 MHz. To realize the simultaneous data collection in the technical and local control room 800 m fiber optic cable was used [69].

6.3 Data Preparation / Preprocessing

Similar procedures used in the previous chapter section 5.4.2 are used here for preprocessing. However, the data was obtained at a sampling rate of one point / sec. For each measuring point it was convenient to use the average of every 10 points such that the data file size can be reduced 10 times. The maximum and minimum values used during the various training sessions in this study are shown in Table 6.1 (a,b).

6.4 MSF Modelling using GENE Approach

One of the difficulties involved in describing the process behaviour of MSF process is the prediction of the dynamic behaviour for the brine level in the various stages. ANN can be used to predict the brine levels from the input information to the network. This black box approach requires the brine level measurments in the different stages of interest to be available and excitation of the system is possible. The current installation is normally equipped with level measurement facilities for the first and last stages. This can be used for the learning process and model validation. A base case study was carried out adopting the GENE approach as the learning algorithm. The inputs to the network are the tapped delayed points for the followings:

1. Brine heater inlet temperature
2. Cooling sea water temperature
3. Sea water temperature
4. Top brine temperature
5. Sea water recirculating flow
6. Sea water make up flow
7. Cooling sea water flow
8. Brine recirculating flow
9. Brine blow down flow
10. Sea water make up temperature to deaerator
11. Brine recirculating inlet temperature
12. Brine level of last stage
13. Distillate flow
14. Brine level of first stage

The network outputs are the current operating points for the last 5 parameters above. Thus the network represent the NARAMAX identification of the system [36]. The procedure used for presenting the tapped delay values in the previous chapter is adopted here.

It is important to identify which of the available data sets used for the learning process will provide good mapping results. In this regard a number of variations of the base case, using different combinations of data sets for training the network with the same structure were also studied. The selection of the data sets combination is based on an engineering understanding of the problem. Next, the variations of the network structure for different combinations of the data sets were also studied.

A parameter t_{SS}, the total sum of squares of the errors between estimated output and desired output, was used as a measure of the performance of the network, where t_{SStr} and t_{SSte} are used to denote the t_{SS} for the training set and the testing set respectively. T_{SS} is the total sum of t_{SStr} and t_{SSte} obtained from presenting all data sets.

To evaluate the weight initialization scheme, a 42-15-15-5 network architecture based on the GENE approach was trained with $dt_{1,2,3,4,5}$ data set. By selecting a low range (RIW) for the weight initialization scheme (see section 4.5.4) and comparing the resultant RMS error achieved with a higher range of RIW. This exercise will stop when no further improvement could be achieved by increasing the range of RIW. When using RIW for the output layer between -0.05 and +0.05, the RMS error converged to a value of 0.145 and could not be reduced with further training. Investigation to use the normalized delta rule or the delta rule for the second hidden layer has revealed the same result. When using RIW for the output layer to be between -0.15 and +0.15, the RMS error converged to 0.028 and 0.038 when the normalized delta rule and the delta rule were adopted for the

second hidden respectively. No further improvement was obtained by increasing the range of the initialization scheme. Therefore, the choice of the initialization scheme used here for the output layer is between -0.2 and +0.2, and the normalized delta rule is used for the second hidden layer.

The learning rate and momentum values used are 0.3 and 0.4 for the first hidden layer and 0.25 and 0.4 for the second hidden layer respectively for the first 3000 epochs. After 3000 epochs the learning rate and momentum values are reduced to 0.15 and 0.2 for the first hidden layer, and 0.125 and 0.2 for the second hidden layer respectively. Training was terminated when one of the following conditions applied:

- Network output RMS error was less than a pre-specified value.
- A specified number of epochs were reached.

6.4.1 Base Case

The study was carried out by using a 42-30-30-5 network based on the GENE approach. Data set of 645 examples from the brine recirculation dynamic test (dt_1) were used for training, and then all other data sets were used for testing. With RIW chosen to be between -0.2 and 0.2 for the outer layer while for the hidden layers the RIW was chosen between -0.01 and 0.01, the network RMS error converged to a value of 0.024 after 4000 epoch. The trained network was tested with all other examples dt_2 to dt_8 resulting in T_{SS} of 64.02. Table 6.2(a) shows t_{sstr} for each output of the network (the output is denoted by O_n, n = 1,2,..5). Table 6.3(a) shows t_{sste} for each output of the network. In some cases the t_{sste} are unacceptably high. When using the standard BP with one hidden layer, the RMS error converged to a value of 0.028 after 4000 epoch with no significant improvement is obtained for t_{sstr} , t_{sste} and T_{SS} as shown in tables 6.2(b) and 6.3(b). It is

obvious, that the performance of the network deteriorates very little with the exception of O_5 (brine level of the first stage).

	O_1 t_{sstr}	O_2 t_{sstr}	O_3 t_{sstr}	O_4 t_{sstr}	O_5 t_{sstr}
dt_1	0.01	001	0.04	0.03	0.06

(a)

	O_1 T_{sstr}	O_2 t_{sstr}	O_3 T_{sstr}	O_4 t_{sstr}	O_5 t_{sstr}
dt_1	0.01	0.01	0.03	0.03	0.06

(b)

Table 6.2: Base case performance for the training sets for each network output using (a) GENE approach, (b) BP approach

	O_1 t_{sste}	O_2 t_{sste}	O_3 t_{sste}	O_4 t_{sste}	O_5 t_{sste}	t_{ss}
dt_2	0.54	0.13	0.03	0.26	8.85	9.81
dt_3	0.45	0.11	0.05	0.14	6.27	7.02
dt_4	0.63	0.25	0.48	0.93	5.76	8.05
dt_5	0.02	0.01	0.13	0.25	0.38	0.79
dt_6	0.01	0.01	0.04	0.05	0.05	0.16
dt_7	0.12	0.23	1.75	0.72	5.3	8.12
dt_8	1.15	0.09	0.62	3.35	24.7	29.91
					T_{ss} =	63.86

(a)

O_1 t_{sste}	O_2 T_{sste}	O_3 t_{sste}	O_4 T_{sste}	O_5 t_{sste}	t_{ss}
0.22	0.43	1.51	0.01	2.89	5.06
0.21	0.39	1.2	0.04	2.15	3.99
0.91	0.81	0.36	1.98	3.97	8.03
0.03	0.04	0.16	0.29	0.37	0.89
0.01	0.01	0.02	0.07	0.1	0.21
0.68	0.57	0.7	3.54	12.31	17.8
0.93	1.63	6.67	2.32	2.60	14.15
				T_{ss} =	50.13

(b)

Table 6.3: Base case performance for the testing sets for each network output using (a) GENE approach, (b) BP approach. The double line border is for $t_{sste} < 2$

6.4.2 Variation 1

Since the base case did not give satisfactory results, tests were made by training the network with a different combination of the data sets. Using the GENE approach with the same network structure as the base case, nine data sets were used for training. In the following $dt_{3,4,5}$ denotes data sets dt_3 , dt_4 and dt_5 are combined in one data set.

- Case 1.1, $dt_{3,4,5}$ is used for training, and then all other data sets were used for testing.
- Case 1.2, $dt_{2,4,5}$ is used for training, and then all other data sets were used for testing.
- Case 1.3, $dt_{1,3,5}$ is used for training, and then all other data sets were used for testing.

- Case 1.4, $dt_{1,3,4}$ is used for training, and then all other data sets were used for testing.
- Case 1.5, $dt_{1,4,5}$ is used for training, and then all other data sets were used for testing.
- Case 1.6, $dt_{1,2,4}$ is used for training, and then all other data sets were used for testing.
- Case 1.7, $dt_{1,2,3}$ is used for training, and then all other data sets were used for testing.
- Case 1.8, $dt_{2,3,5}$ is used for training, and then all other data sets were used for testing.
- Case 1.9, $dt_{1,2,3,4,5}$ is used for training, and then all other data sets were used for testing.

The training was stopped when either the error is less that 0.01 or after 10000 epochs. All cases were converged to a RMS error of less than 0.034. After each training the network is tested with all data sets. T_{SS} results are compared in table 6.4 for the nine cases. It is obvious that when the training examples were increased to cover more than one dynamic test, the performance on the testing set is improved compared with the base case. On the other hand, when the "Low pressure steam flow dynamic test" (dt_4) data is included in the data set for training the network, the performance is further improved.

Case	1.1	1.2	1.3	1.4	1.5	1.6	1.7	1.8	1.9
Training data	$dt_{3,4,5}$	$dt_{2,4,5}$	$dt_{1,3,5}$	$dt_{1,3,4}$	$dt_{1,4,5}$	$dt_{1,2,4}$	$dt_{1,2,3}$	$dt_{2,3,5}$	$dt_{1,2,3,4,5}$
T_{SS}	2.842	1.925	6.734	2.212	2.209	1.925	4.184	7.953	2.542
RMS error	0.008	0.022	0.019	0.023	0.025	0.022	0.022	0.021	0.024

Table 6.4: Variation I case performance for the testing sets for each network output using GENE approach

6.4.3 Variation 2

The optimal number of hidden neurons in a network for a particular mapping is generally unknown beforehand. It would be interesting to see the effect of changing the numbers of hidden neurons. When the number of hidden neurons is increased, the number of connection weights is increased as well. Thus the number of training examples should be increased accordingly. Same training examples used in variation 1 were used. Four cases were studied:

- Case 2.1, a 42-15-15-5 network was used.
- Case 2.2, a 42-20-20-5 network was used.
- Case 2.3, a 42-25-25-5 network was used.
- Case 2.4, a 42-35-35-5 network was used.

The performances of the networks are compared in table 6.5.

For cases 2.1, T_{SS} values are unacceptably high for all combination of training data. On the other hand, the improvement for case 2.2 is not for all combinations of data sets. It seems that a bigger network can be expected to have a better performance than a smaller network. It appears that also due to too many parameters for case 2.4, the performance starts to deteriorate (compared to the base case in table 6.4) for the same combinations of data sets. Therefore the base case hidden neurons number used can be considered as the optimum.

Case	2.1		2.2		2.3		2.4	
Training data	T_{ss}	*RMS*	T_{ss}	*RMS*	T_{ss}	*RMS*	T_{ss}	*RMS*
$dt_{3,4,5}$	13.76	0.027	3.59	0.025	2.416	0.008	2.498	0.008
$dt_{2,4,5}$	7.982	0.026	2.772	0.024	4.298	0.022	2.134	0.02
$dt_{1,3,5}$	24.6	0.025	8.457	0.091	7.583	0.022	5.977	0.025
$dt_{1,3,4}$	11.9	0.031	2.894	0.021	2.553	0.023	2.162	0.023
$dt_{1,4,5}$	10.98	0.031	2.557	0.02	3.453	0.023	3.591	0.024
$dt_{1,2,4}$	15.27	0.033	4.2	0.02	2.057	0.025	2.005	0.024
$dt_{1,2,3}$	20.32	0.014	7.964	0.009	4.787	0.022	7.298	0.024
$dt_{2,3,5}$	20.98	0.034	9.584	0.023	8.496	0.021	9.086	0.028
$dt_{1,2,3,4,5}$	16.66	0.033	2.797	0.028	2.967	0.027	3.37	0.028

Table 6.5: T_{SS} *results after training using GENE approach for different number of hidden neurons*

6.4.4 Variation 3

Further investigations using different hidden number of neurons and with different data sets confirms the improved results of the *GENE* approach as compared to the standard BP.

	h1	*h2*	*w*	*L*	*d*	*RMS*	T_{ss}
1	34	43	0.25	GENE	$dt_1,dt_2,..dt_7$	0.021	16.25
2	34*	43	0.25	GENE	$dt_1,dt_2,..dt_7$	0.03	4.53
3	15*	10	0.25	GENE	$dt_1,dt_2,..dt_7$	0.038	4.94
4	45*	45	0.25	GENE	$dt_1,dt_2,..dt_7$	0.022	2.79
5	45*	45	0.25	GENE	$dt_{1,2,3,4,5}$	0.019	2.215
6	45*	45	0.25	GENE	$dt_{1,2,3,4,5}$	0.019	2.65
7	45*	45	0.25	GENE	$dt_{1,2,3,4,5}$	0.028	2.85
8	45*	45	0.25	GENE	$dt_{1,2,4}$		3.469
9	30*	-	-	BP	$dt_{1,2,4}$		2.97
10	30*	-	-	BP	$dt_{1,2,3}$		40.49
11	45*	45	0.25	GENE	$dt_{1,2,3}$		8.02
12	30*	-	-	BP	dt_4		13.45
13	35*	35	0.2	GENE	dt_4		10.31

Table 6.6: A comparison for Tss & RMS results when the network is trained using GENE approach & the standard BP learning algorithm

6.5 Learning Response of the Control System due to External Disturbances for MSF Plants

In the last chapter, it has been shown that the required *function approximation* for any mapping problem can be achieved using ANN through the learning process with the aim to minimize the network error by modifying the weights. However, potential problems, which are likely to affect practical applications, are that the learning process may be seriously plagued by the presence of local minima in the cost function and the long training time. Learning is normally achieved based on the knowledge of local error associated with each hidden neuron and not the network error [62], [36], [30]. For this, most of the reported success is based on the assumption that sufficient and rich data are available for the training to capture the required function approximation. But in most of the practical cases it is generally required to train the network with partial data and also, if trained with additional new data. It should not forget the previous training. Researchers have modified either the weight update rule [27], or employed a new form of objective function [70]. In this section, the MFN network using the *GENE* approach is studied and compared to the standard BP learning algorithm by developing a nonlinear empirical multi-controller structure for MSF plants (figure 6.1 shows a simplified block diagram of the sructure). The data obtained for the previous section (dt_1, dt_2 , dt_3 , dt_4 , dt_5 , dt_6 , dt_7) are used for training, then the problem of finding set of weights for a given data set (dt_n) can be solved using ANN, but the additional learning using another data set (dt_m) for the resultant network will have either one of the following effects on the network:

1) the network will learn the new data, but loses part of the previous function learnt.; or
2) the network will not converge as the two directions are opposite or higher order is required.

The study was carried out with the inputs to the network being the tapped delay points for all the variables listed in tables 6.1 and 6.10 (this includes the controller set-point, control valve position, regulated variable for the seven control loops and additional monitoring points), excluding the outputs of each controller which are used as the network outputs. Instead of feeding all past data to the network, the following equation is used for calculating the tapped delayed values:

$$D = STD\,[(S-S_{max})/S_{span} + (V-V_{max})/V_{span} + (C-C_{max})/C_{span}]. \qquad (6\text{-}1)$$

Where STD is the standard deviation, S is the set point, V is the control valve position, and C is the controlled variable.

The parameter T_{SS} used in the previous section was used here again as a measure of the performance of the network.

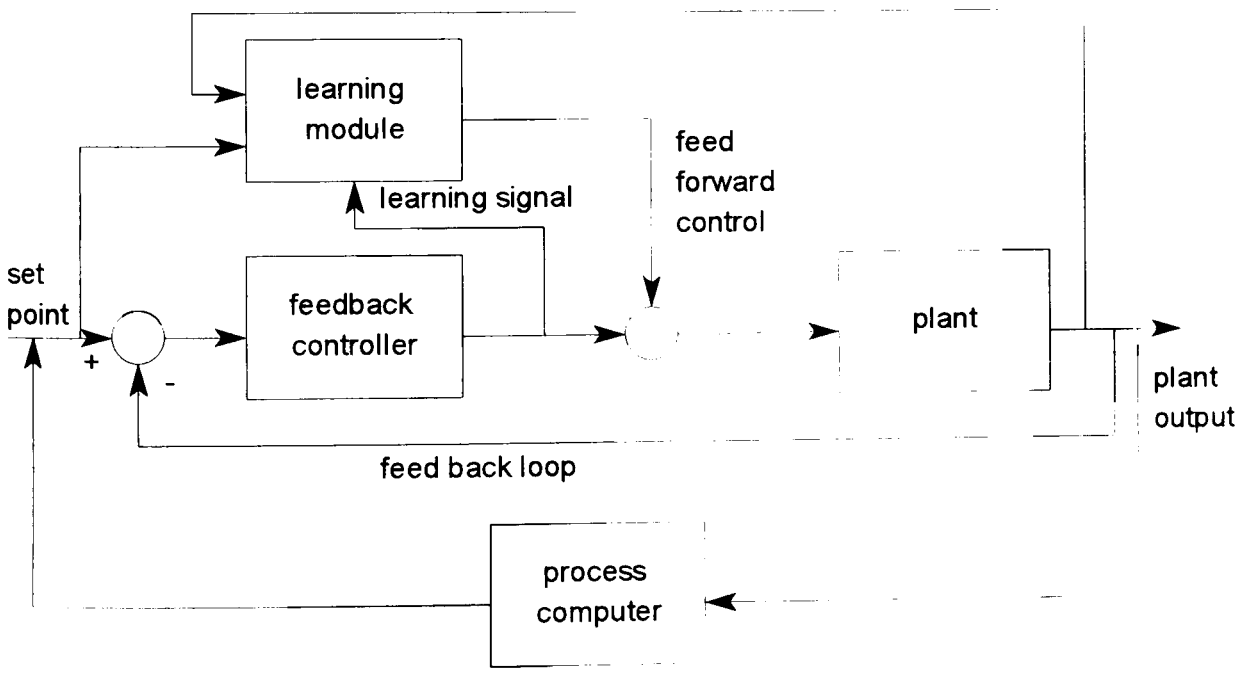


Figure 6.1: a simplified nonlinear empirical multi-controller structure for MSF plants

6.5.1 Training using the Standard BP Learning Algorithm

Firstly, we used the standard BP learning algorithm with one hidden layer network and found that when the network is subject to further training with new data, a drift occurs when testing the previously trained data. This is due to the static feature of the single hidden layer. No improvement could be achieved by adding more hidden neurons. It was found that by using the delta rule for the output and hidden layers instead of the normalized commutative delta rule, the RMS error for the network is converging to a considerably lower value. The learning rate and momentum values used were 0.05 and 0.4 for the output layer, and 0.3 and 0.4 for the hidden layer respectively for the first 5000 epochs. After 5000 epochs the learning rate and momentum values are reduced to 0.0355 and 0.2 for the output layer, and 0.15 and 0.2 for the hidden layer respectively. Data set of 1028 examples from low pressure steam temperature dynamic test, makeup flow dynamic test and sea water supply temperature dynamic test ($dt_{5,6,7}$) was used for the initial training for 4000 epochs. Then the network is subject to further training with data set of 547 examples from brine recirculation dynamic test and brine level dynamic test($dt_{1,2}$). The network is then further trained with 561 examples from distillate level dynamic test and low pressure steam flow dynamic test ($dt_{3,4}$). All available data sets ($dt_{1,2,3,4,5,6,7}$) were used for studying the network performance. Four cases were studied:

- Case 1.1, a 107-130-7 network was used.
- Case 1.2, a 107-110-7 network was used.
- Case 1.3, a 107-90-7 network was used.
- Case 1.4, a 107-70-7 network was used.

The resultant T_{SS} for each output of the network are shown in table 6.7, where (f1) denotes results after first training, (f2) denotes results after second training and (f3) denotes results after third training. The values of T_{SS} shown are neither acceptable (high value) nor improving with the additional training conducted by another set of examples. However, the best result is for case 1.3, for which a comparison will be made to the GENE approach.

		Tss(O1)	*Tss(O2)*	*Tss(O3)*	*Tss(O4)*	*Tss(O5)*	*Tss(O6)*	*Tss(O7)*
Case 1.1	f1	1468.6	450.1	4883	2831.7	8185.2	13528.9	7171.6
	f2	1727.5	348.7	2812	445.2	1589.3	3000.6	1084.9
	f3	1660.8	476.7	5934.9	838.4	1461.1	4672.0	630.8
Case 1.2	f1	11308.1	151.2	3011.5	1210.8	6256.6	16856.3	458.2
	f2	2111.7	456.3	6539.8	872.5	1520.3	3687.8	418.2
	f3	1936.2	492.6	12179.2	695.4	1495.9	6273.8	785.1
Case 1.3	f1	3019.6	669.9	7690.3	1139.7	6612.1	15878.1	781.4
	f2	5960.8	515.6	5532.4	597.4	1308.8	3123.3	1023.7
	f3	3651.3	399.1	5837.6	506.6	1027.1	8126.7	1837.2
Case 1.4	f1	3450.2	335.5	5188.9	3999.7	8785.1	20326.5	124.5
	f2	4480.6	551.7	7274.6	1518.1	1382.5	2767.9	1055.8
	f3	3783.5	535.8	10002.5	1112.9	1176.6	2059.2	1531.9

Table 6.7: Tss values for each BP network output after first (f1), second (f2) and third (f3) training of different examples.

6.5.2 Training using GENE Approach

All simulations presented in this study utilize the same network architecture. In particular, the network has 107 inputs, 7 outputs, 60 nodes in the first hidden layer and 30 nodes in the second hidden layer.

To evaluate the RIW scheme, a 107-60-30-7 network architecture based on the GENE approach was trained with $dt_{1,2,3,4,5}$ and the following cases for random initialized weight schemes are considered:

- Case 2.1, the network is trained using RIW in the range [-0.1 , 0.1].
- Case 2.2, the network is trained using RIW in the range [-0.05 , 0.05]
- Case 2.3, the network is trained using RIW in the range [-0.01 , 0.01]

Table 6.8 shows the RMS values for each layer after 5000 epochs. These values could not be reduced with further training. Investigations using the normalized delta rule or the delta rule for the second hidden layer have revealed the same result.

	output layer	first hidden layer	second hidden layer
Case 2.1	0.123953	0.016297	0.19504
Case 2.2	0.030918	0.004721	0.004792
Case 2.3	0.037433	0005542	0.006425

Table 6.8: RMS error values for each layer after 5000 epochs.

When using RIW between -0.05 and +0.05 for the output layer, the RMS error converges to 0.030918, 0.004721 and 0.004792 for the output, first and second hidden layers respectively. No significant improvement was obtained by decreasing the RIW range. Therefore, the choice of the RIW scheme for the output layer is between -0.05 and +0.05, and the normalized delta rule is used for the second hidden layer.

The learning rate and momentum values used are 0.3 and 0.4 for the first hidden layer and 0.25 and 0.4 for the second hidden layer respectively for the first 10000 epochs. After 10000 epochs the learning rate and momentum values are reduced to 0.15 and 0.2 for the

first hidden layer, and 0.125 and 0.2 for the second hidden layer respectively. The same procedure for the first, second and third training used for the BP is used here. However, the first training was made for 25000 epochs, while the second and the third training are made for 5000 epochs.

The resultant T_{SS} for each output of the network are shown in table 6.9 for the following cases:

- Case 3.1,Gene approach is used during the training, and RIW scheme used for the output layer is between -0.05 and +0.05, and the normalized delta rule is adopted for the second hidden layer.
- Case 3.2, same training procedure for case 3.1, but the RIW scheme used for the output layer is between -0.1 and +0.1.
- Case 3.3, the standard BP algorithm with single hidden layer and 90 neurons are used during the training. The delta rule is adopted for the output and hidden layer.

		Tss (O1)	*Tss(O2)*	*Tss(O3)*	*Tss(O4)*	*Tss(O5)*	*Tss(O6)*	*Tss(O7)*
Case 3.1	f1	4555.5	1198.6	7649.2	2184.8	8738.5	6944.7	973.5
	f2	3728.2	291.8	3846.9	1327.1	1811.3	1279.3	271.5
	f3	3525.4	361.1	4327.8	458.4	1262.3	1067.1	159.1
Case 3.2	f1	20809.1	2970.1	10679.9	1470.1	7889.8	40192.1	427.2
	f2	1617.6	364.1	6880.3	1657.4	1806.7	3795.6	1904
	f3	1392.3	332.1	5790.5	1510.3	1411.1	4153.2	1514.5
Case 3.3	f1	781.3	695.7	2382.5	917.1	7018.2	17117.4	910.7
	f2	1438.1	140.3	2747.1	1156.2	2107.6	2915.9	239.1
	f3	1555.5	206.5	4520.1	1320.6	1580.3	10231.3	360.9

Table 6.9: Tss values for each network output after first, second and third training.

The best results are shown when using the GENE approach for case 3.1, for which the additional training conducted using new examples has improved the *Tss* values for each output of the network. When using the standard BP with one hidden layer (Case 3.3), the Tss value does not improve for all network outputs except for output 2,7 where it is improves after the second training, but deteriorates after the third one. It is obvious, that the performance of the network using the GENE approach is far better than the BP for learning the additional data while retaining the previously learned data. The GENE approach is shown to be robust for the learnt analog data. The predicted network output is compared to the reference data for the three cases and is shown in figures 6.6, 6.7 and 6.8 after learning with the first, second and third data set respectively for one output only.

6.5.3 Error Value Analysis for Output & Hidden Layers

We have discussed in chapter 4 the requirement of unique error transformation when the error is back propagated (transformed) to the hidden layer, such that no change to the gradient search direction during the training. This will yield an increase to the probability of reaching a global solution for the errors. To illustrate this, the RMS error values for the three cases are plotted in figure 6.2. Case 3.3 is shown to provide the least RMS error values during the entire training process. However, this does not mean that it is the best case. For each case, the error values from each layer (output & hidden) during the learning process are plotted in figures 6.3, 6.4 and 6.5. It is obvious that only for case 3.1, the same error pattern is transformed to the hidden layers, i.e, when the error value increases in the output layer during the additional training phase, the error values for the hidden layers follows the same increase, and visa versa. While for cases 3.2 & 3.3, the error values are not following the same transformation pattern, but they are changing abruptly for the output layer. This is due to the change that occurs in the direction of the gradient

during the learning process. For case 3.2, this is due to high initialized weight value selected, such that the node activation for the tanh function will not be in the linear region, while for case 3.3 (standard BP), this is due to the change of gradient direction for each iteration to update the weight values between the output and hidden layers after each iteration.

As discussed in chapter 4, for the initialized weight values in the range [-0.05, +0.05], and using the tanh function as the activation for the nodes, the activation will lie in the linear region and far from the saturation region. This means that the hidden nodes will be trained to reduce the network output error. When the learning process continues for additional data, the weight updates are shown to be in the direction of reducing the network error but not reducing the local error, i.e., retaining the function approximation requirement. As both data sets are related to the same process, so the learning process will capture the function approximation of both data sets and not just interpolate each one at each training cycle. For higher initialized weight values, the hidden layers are shown not to follow the same network output error characteristic, and this means that it does not follow the same gradient direction of the output layer.

6.6 Conclusion

To a large extent the validation results demonstrate that the developed empirical multi-controller structure is indeed capable of generalizing the process dynamics during operational transients, from only a limited training set. The performance is increased when trained with another test data (another loop) and good results are obtained on all the available test data.

The benefit from the additional learning capability is that as the operating point changes from one state to another, the learning controller has not forgotten what has been learnt earlier and it can be improved upon. The learning controller builds up a nonlinear model of all desired control surfaces. This requires the learning module, which is temporally stable, to learn about one area in the input space which affects minimally the knowledge stored in different regions.

The network is initially trained using the transient test data obtained from exciting one of the manipulated variables. When using the standard BP, a significant long training time is required for convergence. The adjustment for the learning rate and the momentum was a tedious job, even though the developed structure was found valid in its entire operating envelope. When using the *GENE* approach, two main distinctions of this approach are : - (1) its ability to generalize and when new test data is introduced the performance of the structure is improved for the test, (2) very fast convergence as compared to the standard one. The resulting trained neural network is shown to intelligently generalize all the available test data and reduce the uncertainty associated with the probability for convergence to the global minimum. Simulation results indicate a superior convergence speed for the *GENE* approach and robustness for analog data. Robustness of the network arises when the network is trained with new data. The network will not forget the previously learned data in order to learn the new data.

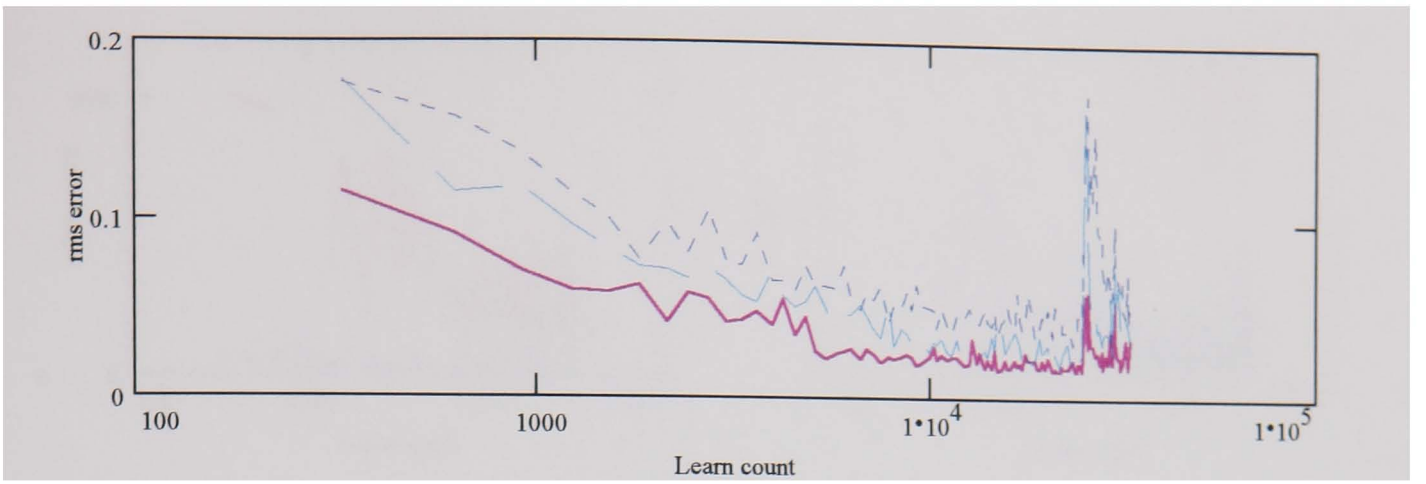


Figure 6.2: RMS Error values during network learning using the BP (solid), Gene with initialized weight (-0.1,0.1) (dot) and Gene with initialized weight (-0.05,0.05) (dash) respectively.

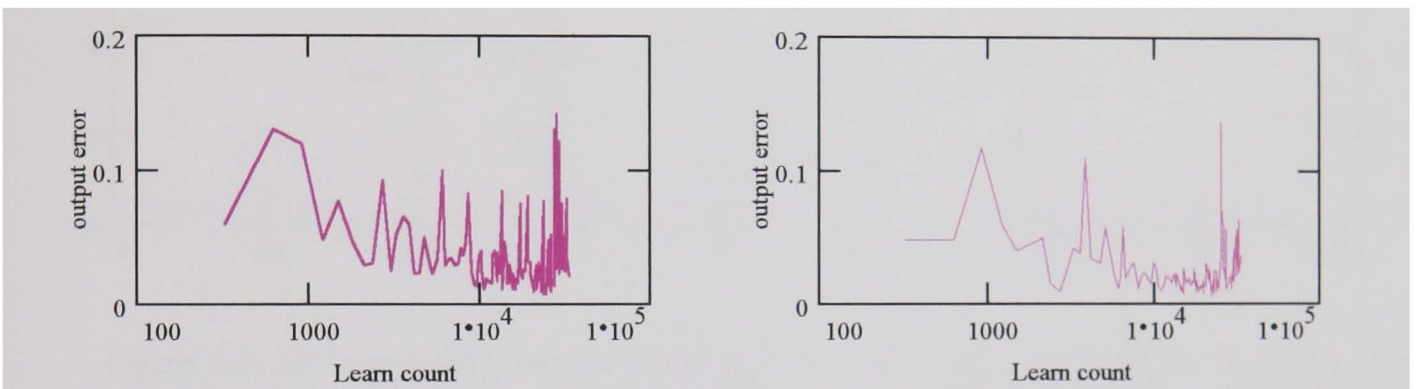


Figure 6.3: Error values during network learning for output layer using Gene approach with initialized weight (-0.1,0.1) (left) and using Gene approach with initialized weight (-0.05,0.05) (right).

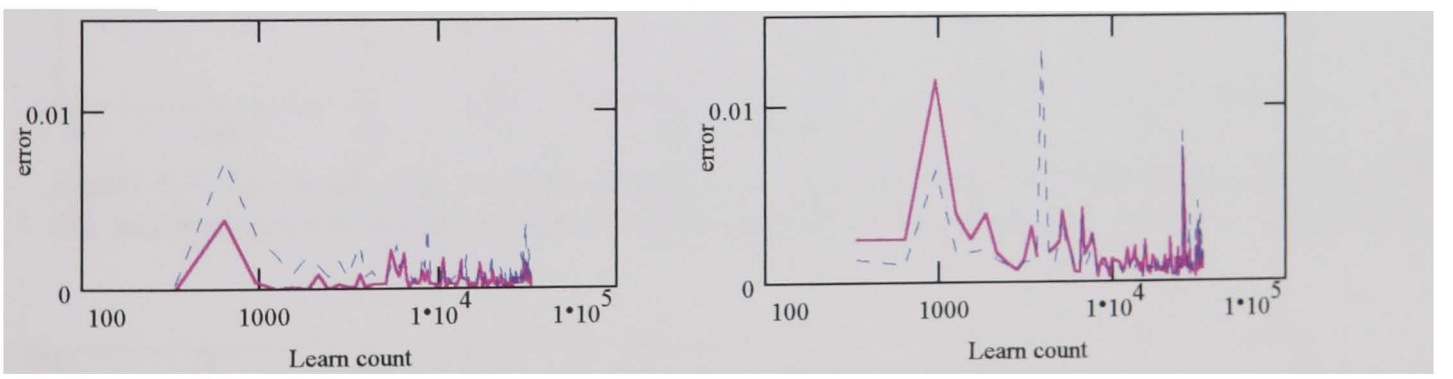


Figure 6.4: Error values during network learning for hidden layers using Gene approach with initialized weight (-0.1,0.1) (left) and using Gene approach with initialized weight (-0.05,0.05) (right).Solid line for hidden layer # 1.

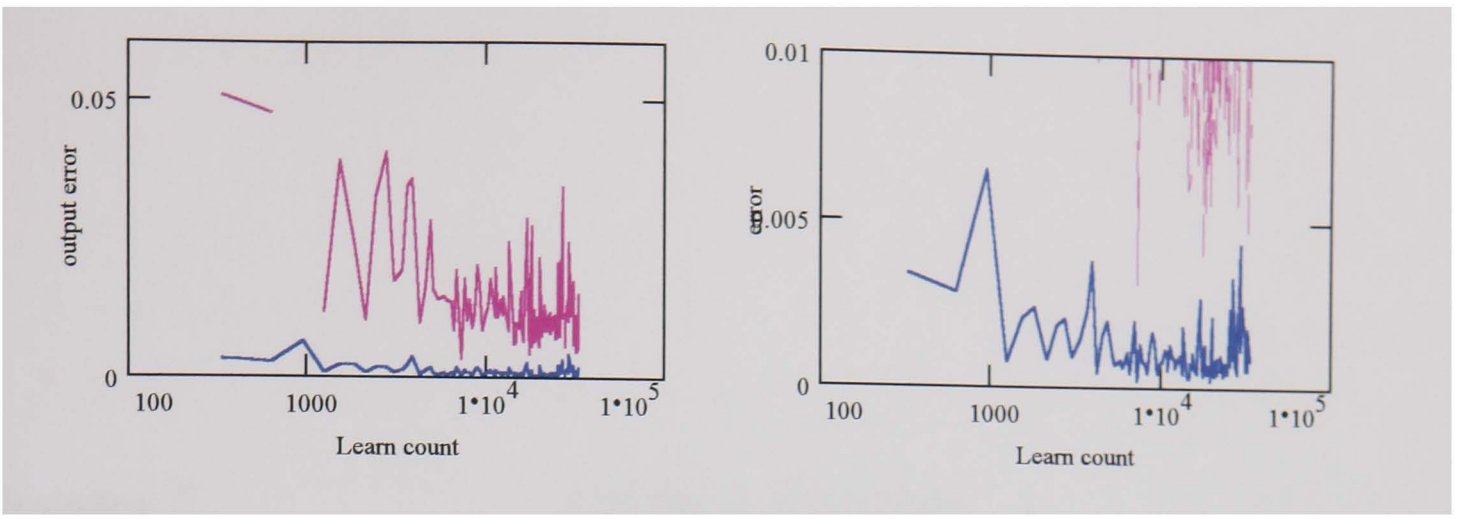


Figure 6.5: Error values during network learning using the BP for output (left) and hidden (right) layers respectively.

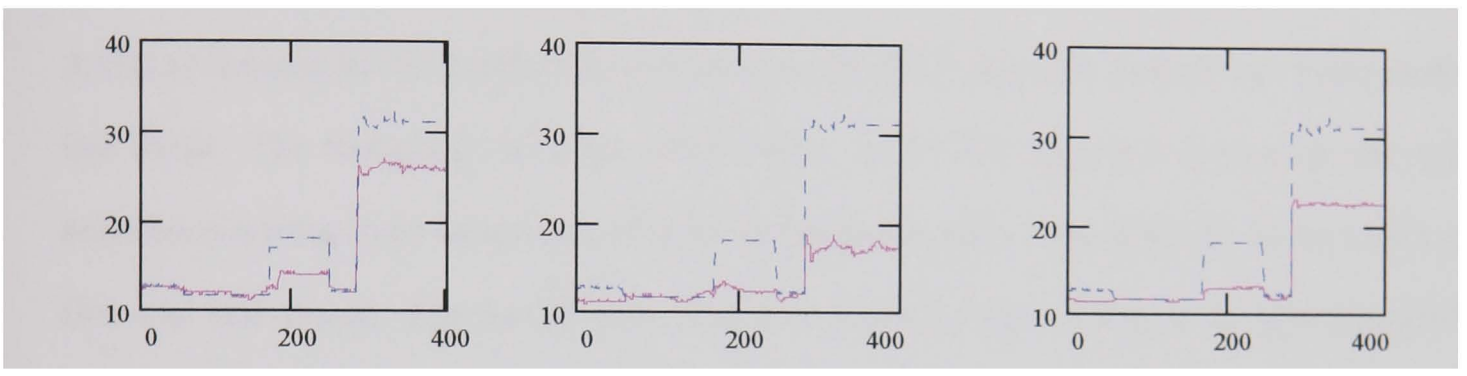


Figure 6.6: Test result after first training for cases 3.1,3.2 and 3.3 from left to right. The solid line and the solid line are for the network prediction and the corresponding reference respectively.

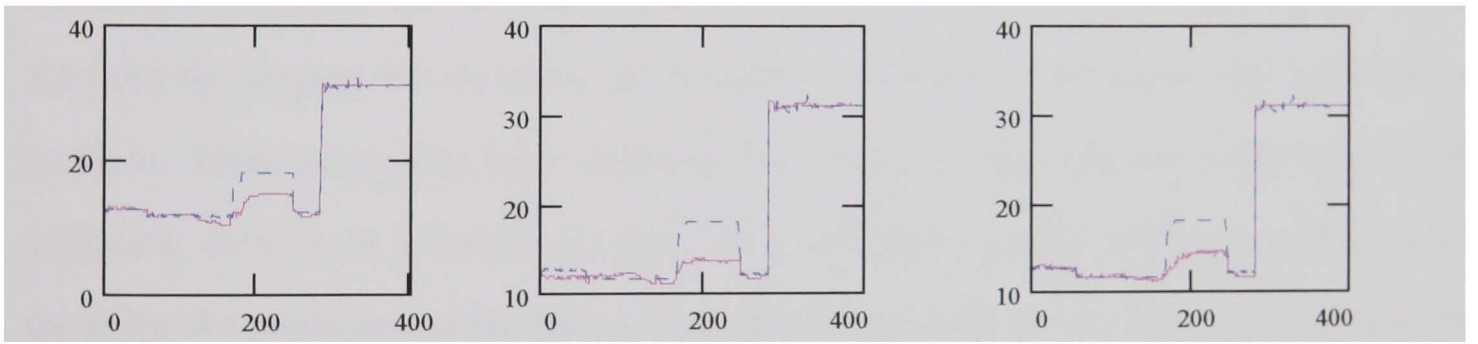


Figure 6.7: Test result after second training for cases 3.1,3.2 and 3.3 from left to right. The solid line and the solid line are for the network prediction and the corresponding reference respectively.

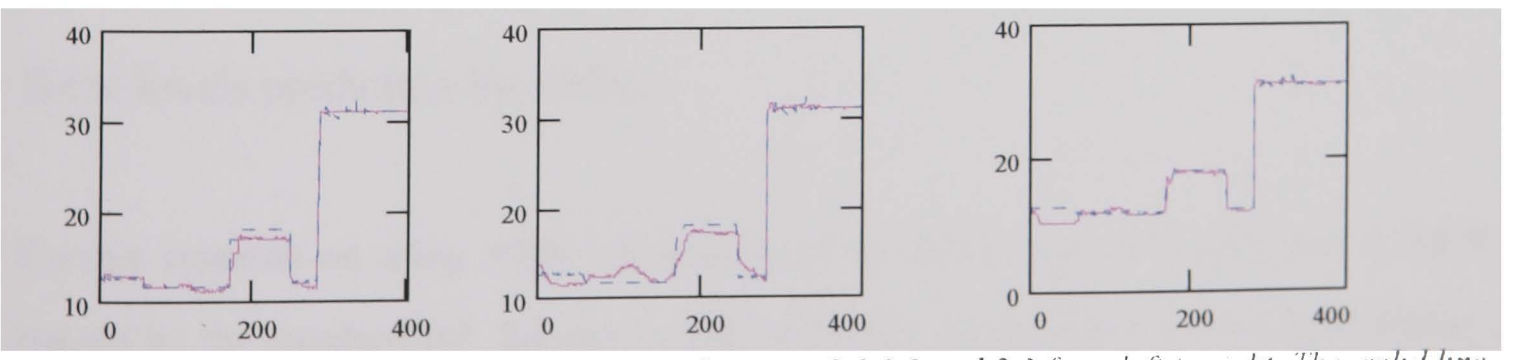


Figure 6.8: Test result after third training for cases 3.1,3.2 and 3.3 from left to right. The solid line and the solid line are for the network prediction and the corresponding reference respectively.

Chapter 7 CONCLUSIONS

A fast technique in the prediction and analysis of MSF dynamic system has been studied in this thesis. The technique uses the ANN based on *GENE* approach to predict the process behaviour during load variations of MSF desalination plant including the brine levels in the first and last stages. The neural networks are capable of providing a nonlinear model that explains the relationship between input-output variables. The two main distinctions of this learning algorithm (*GENE*) are: (1) no saturation for network nodes is exhibited to avoid the ultimate paralysis of the entire MFN during learning, (2) Global convergence can be achieved by keeping the direction of the gradient constant, this avoids the local minima problem. These properties have improved the ability of the network to be trained with additional data in an effective manner. The additional training ability is studied in this thesis for the prediction of the various controllers response due to external disturbance on the MSF desalination plant.

Brine levels prediction by ANNs

Current research on using ANN for brine level prediction is mostly restricted to MFN trained by the standard BP. Subject to the availability of data from empirical simulators, successful simulations have been achieved. For MSF plants, persistent excitation is

required to be performed on the system, particularly to the steam/top brine temperature. However, two modes of control have been studied. One is the smooth load variation between minimum and maximum load condition, the other is a disturbed plant mode of operation. For both cases, *GENE* approach for ANN is shown to give satisfactory results in prediction of the process behavior including the brine levels as compared to the standard BP. For 16 stages MSF plant, ANN with all inputs/outputs can be trained easily and perform accurate predictions of the brine levels. In fact it is found that for the dynamic mode using the *GENE* approach subject to additional training using additional training data, reasonable, accurate and improved prediction of controllers response including the brine level controller can be achieved.

Although some preliminary successful results were reported in this thesis, some inherent difficulties exist with the use of ANNs:-

- It takes very long time to generate training examples.
- The predicted brine level and the controller response can be pessimistic or optimistic.

Future Work

Future work for the development of these techniques would be to study the application of GENE approach for ANN to improve the performance and the reliability of MSF plant. Additionally to compare the GENE approach to other types of neural networks such as the recurrent networks.

Appendix I

Mathematical models used to assist the set point sequence change at the plant in Abu Dhabi (U.A.E.) for 16 stage (13 recovery + 3 reject) MSF units are described in the following:

Three mathematical models are adopted;

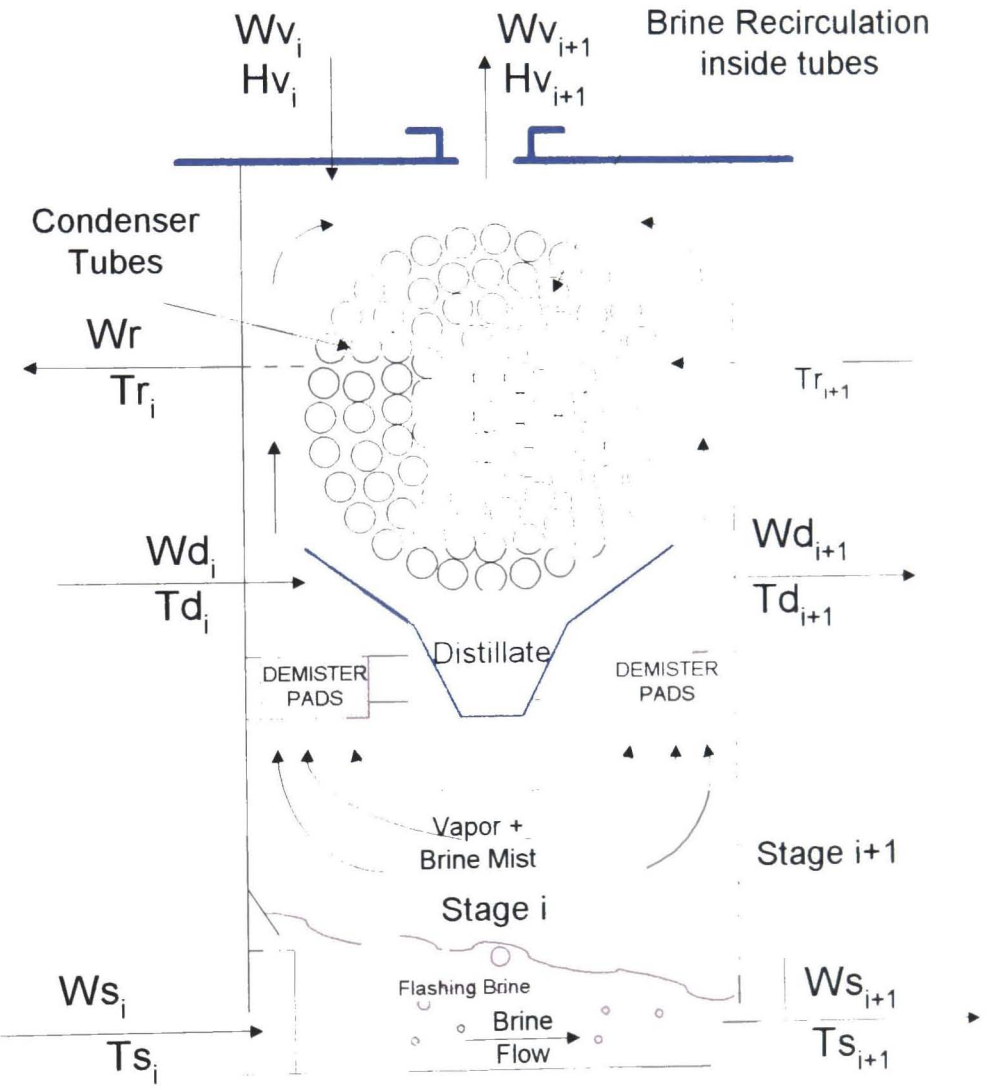
- The first model is used for fouling factor calculation, by dividing the evaporator into groups of three stages (five for the recovery section, one for the reject section). Using the temperatures measured at the tube bundle side for each three stage (6 + 1 on the brine heater), with the brine temperature for each stage, with the distillate space temperature of the last stage of each group, and using the heat exchange equation for each group. By this it is possible to calculate the brine temperature at the tube bundle inlet, knowing the temperature of the condensing steam and the brine temperature at the tube bundle outlet. The measurement is made for each group of three stages, the model is used to calculate the outlet temperature from the group by applying 3 consecutive equations and comparing the result with the measured value. The model is solved iteratively using Newton method with an initial guess or later using the last fouling factor calculated.

- The second model is a short cut model used for simplified calculation of the set points. The model calculates an approximate set points for the TBT and the recycle brine flow rate values that must be set to obtain the requested production (fixed running conditions); and the values of other process variables as flow rate and temperatures, using the energy and mass balance equations and the overall heat exchange equations. The recovery and the reject sections are assumed as single blocks. The method used is based on a repetitive calculation and on the use of the operation curves

- The third model is a detailed mathematical model for each stage of the desalination unit. The variation of the fouling degree is input from the first model. A description of the equations involved in the calculation is in the following.

Mathematical model equations

The mathematical model is based on the resolution of a system of linearized equations, as follows (see fig below):



Overall heat balance

Recovery section:

$$W_{s_i}.cps_i.T_{s_i} - W_{s_{i+1}}.cps_{i+1}.T_{s_{i+1}} + W_{d_i}.cpd_i.T_{d_i} - W_{d_{i+1}}.cpd_{i+1}.T_{d_{i+1}} - W_{r_i}.cpr_i.T_{s_i} + W_{r_i}.cpr_{i+1}.T_{s_{i+1}} = W_{v_{i+1}}.H_{v_{i+1}} - W_{v_i}.H_{v_i}$$

Reject section:

$$W_{s_{i+1}}.cps_{i+1}.T_{s_{i+1}} - W_{s_{i+2}}.cps_{i+2}.T_{s_{i+2}} + W_{d_{i+1}}.cpd_{i+1}.T_{d_{i+1}} - W_{d_{i+2}}.cpd_{i+2}.T_{d_{i+2}} - W_f.cpr_{i+1}.T_{s_{i+1}} + W_f.cpr_{i+2}.T_{s_{i+2}} = W_{v_{i+2}}.H_{v_{i+2}} - W_{v_i}.H_{v_i}$$

Heat exchange equation of the upper part of the stage

Recovery section $$W_r . cpr_i . T_{r_i} - W_r . cpr_{i+1} . T_{r_{i+1}} = U_i . A_i . \frac{T_{r_i} - T_{r_{i-1}}}{\ln \frac{T_{c_i} - T_{r_{i+1}}}{T_{c_i} - T_{r_i}}}$$

Reject section $$W_f . cpr_{i+1} . T_{r_{i+1}} - W_f . cpr_{i+2} . T_{r_{i+2}} = U_i . A_i . \frac{T_{r_{i+1}} - T_{r_{i-2}}}{\ln \frac{T_{c_i} - T_{r_{i+2}}}{T_{c_i} - T_{r_{i+1}}}}$$

with the following boundary conditions:

in the first stage $T_{s_1} = T_{\max}$

in the last stage $T_{r_{22}} = T_{sw}$

in the last recovery stage and in the first reject stage $T_{s_{18}} = T_{s_{19}}$

in the recovery stage in the last reject stage $T_{r_{18}} = T_{s_{22}}$

where

T_s Flashing brine temperature

T_c distillate temperature

T_r recirculation brine (feed water) temperature in tubes

W_s flashing brine flow rate

W_r recirculating brine flow rate

W_f feed water flow rate

W_v steam flow rate

H_v steam enthalpy

U overall heat exchange coefficient

A stage heat exchange surface

cps, cpd, cpr specific heat capacity

References

1. Dimitris, C.P. and L.H. Unger, 1992: "*A Hybrid Neural Network - First Principle Approach to Process Modelling*", AIChE Journal Vol. 38, No. 10, pp. 1499-1511.
2. Al-Gobaisi, D.M.K., A.S. Barakzai, and A.M. El-Nashar: *An Overview of Modern Strategies for Optimizing Thermal Desalination Plants*; Desalination 84: 30-43 Elsevier Science Publishers B.V.Amsterdam.(1991)
3. Al-Gobaisi, D.M.K., A. Hassan, G. P. Rao, A. Sattar, A. Woldai, and R. Borsani, *Towards improved automation for desalination processes, Part I: Advanced control*, Desalination 97:469-506. Elsevier Science Publishers B.V. Amsterdam. (1994)
4. Rao, G.P., D.M.K. Al-Gobaisi, A. Hassan, A.Kurdali, R. Borsani and M. Aziz, *Towards improved automation for desalination processes, Part II: Intelligent control,* Desalination, 1994. 97:507-528. Elsevier Science Publishers B.V. Amsterdam.
5. El-Hawary, M.E. "*Artificial Neural Networks and Possible Application to Desalination*", Desalination 1993. 92:125-147. Elsevier Science Publishers B.V. Amsterdam.
6. Rumelhart, D. E., G.E. Hinton, and R.J. Williams," *Learning internal representations by error propagation, in Parallel Distributed Processing, Vol. 1: Foundations*, D. E. Rumelhart , J. L. McClelland and the PDP research group, Eds. Cambridge, MA: MIT Press, 1986. 318-362.
7. Beamer, J.H., and D.J. Wilde; *The simulation and Optimization of a Single Effect Multistage Flash Desalination Plant* ; Desalination 9 (1971), 259-275
8. Barba, D., G. Linzzo, and G. Tagliferri; *Mathematical Model for Multiflash Desalting Plant Control*; 4th Int. Symp. on Fresh Water from Sea Vol 1 (1973), 153-168
9. Rautenbach, R. and H. Buchel, *Modular programe for design and simulation of desalination plants*, Proc. 7th Int. Symp. fresh water from the sea, 1 1980. 153-161
10. Omer, A, *Simulation of MSF desalination plants*, Desalination, 45, 1983, 65-67.
11. Medani, M, M.Soliman and A. Helal, *Computer simulation of desalination plants in Saudi Arabia*, Proc. 7th Int. Symp. frash water from the sea, 1, 1980, 85-98.
12. Helal, A., M. Medani and M. Soliman, *A tridiagonal matrix model for multi stage flash desalination plants*, Comp. Chem. Eng. 10(4), 1986, 327-342
13. Amundson N.R. and A.J. Pontinin, *Ind. Engng Chem.* 50, 730 (1958)
14. Wang, J. and G. Henke, *Tridiognal Matrix for distillation*, Hydrocarbon Processing, 45(8), 1966, 155-163.
15. Husian, A. " *Introduction to column design*", Chap. 5 in Distillation Dynamics and control by P.B. Deshpande, Instruments Society of America, 1980.
16. Husain, A., A. Woldai, Adel Al-Radif, A. Kesou, R. Borsani, H. Sultan, P.B. Deshapnde, *Modelling and simulation of multistage flash (MSF) desalination plant.*

Proceedings of the IDA and WRPC world conference on desalination and water treatment. Vol. III, 119-151, (1991).

17. Glueck, A.R. and W. Bradshaw : *A Mathematical Model for Multistage Flash Distillation Plant*; 3rd International Symposium of Fresh Water from Sea, Vol 1, 95-108 (1970)
18. EL-SAIE, M.H.A., M.F. Farrag, and M.H. Erfan : *Computer supervisory Control of Multistage Flash Evaporator*; Desalination, 55 (1985), 107-118.
19. Fareigh, Darwish M., and S. Arazzini: *Description and Mathematical Model of a Large MSF Desalination Plant in SCADA configuration*, Desalination 55 (1985), 91-106.
20. Fumagalli B. and E. Ghiazza, *Mathematical Modelling and Expert Systems Integration for Optimum Control Strategy of MSF Desalination Plants*, Regional Symposium on Water Desalination "DESAL" 92, AL-AIN, U.A.E.
21. Wiener N. 1948: "*Cybernetics: or Control and Communication in the Animal and the Machine*", MIT Press, Cambridge, MA.
22. Aström K.J., and B. Wittenmark 1989. *Adaptive Control,* Addison Wesley, Reading, MA.
23. Widrow B. 1987: "*The Original Adaptive Neural Net Broome-Balancer*", Proc. IEEE Int. Sympo. Circuits and Systems, pp. 351-357
24. Hornik K. "*Approximation Capabilities of Multilayer feed forward networks*", Neural Networks, Vol. 4, pp. 251-257, 1991.
25. Psaltis D. , A. Sideris, and A. Yamamura, "*Neural Controllers*" IEEE Int. Conf Neural Networks, Vol. 4, 1987 pp. 551-558
26. Tariq Samad, "*Neural Networks for Control-An Overview*", ISA, 1991 paper #91-0560-1054 0032/91/ 1939-1946
27. Nguyen D. and B. Widrow, "*Neural Networks for self-learning control system*", in Proc. International Journal of Control, 1991, vol.54, No.6, pp. 357-363.
28. Jordan M.I. and R.A.Jacobs, "*Learning to control an unstable system with forward modelling*, "in Advances in Neural Information Processing Systems, vol. 2, D. S. Touretzky, Ed. San Mateo, CA: Morgan Kaufmann,1990, pp. 324-331.
29. Bhat N.V., P.A. Minderman, T. McAvoy and N. S. Wang, *Modelling chemical process systems via neural computation*. IEEE *Control Syst. Meg.* 10, 24-30 (1990).
30. McAvoy T., N. S. Wang, S. Naidu, N. V. Bhat, J. Gunter and M. Simmons, *Interpreting biosensor data via back propagation.* In *Int. Joint Conf. on Neural Networks,* pp. I-227-I233 (1989).
31. Hoskins J. and D. M. Himmelblau, *Artificial neural network models of knowledge representation in chemical engineering*. Computer chemical Engineering 12, 881-890 (1988).
32. Narendra K.S. and K. Parthasarathy. *Identification and control of dynamical systems using neural networks*, IEEE *Trans. Neural Networks,* Vol. 1, pp. 4-27, Mar. 1990.
33. Levin A.U. and K.S. Narendera. "*Control of non-linear dynamical systems using neural networks: Controllability and Stabilization*", IEEE *Trans. Neural Networks,* vol. 4, pp. 192-206, Mar. 1993.

34. Saint-Donat J., N. Bhat and T.J. McAvoy 1994: "*Neural Net Based Model Predictive Control*", in "Advanced in Intelligent Control", Eds Harris C.J., Taylor and Francis, London, Chapter 8.
35. Chen, S. , S.A. Billings and P.M. Grant, *Non-linear system identification using neural networks*, Int. J. Control, vol. 51, no. 6, pp. 1191-1214, 1990
36. Leontaritis, I. J. and S. A. Billings, "*Inputs-output parametric models for nonlinear system. Part 1: Deterministic nonlinear systems. Part 2: Stochastic nonlinear systems*," Int. J. Control, vol. 41, pp. 303-344, 1985.
37. Werbos P.J., T. McAvoy , T. Su 1992: "*Neural Networks, System Identification and Control in the Chemical Process Industries*", in "Handbook of Intelligent Control", Eds D.A. White, D.A Sofge and Van Nostrand Reinhold, NY, Chapter 10.
38. Stengel R.F. 1992: "*Intelligent Flight Control Systems*", Proc. IMA Conf. Aero Veh Dyn. and Control, Cranfield Institute of Technology, UK.
39. A F Abdelbary, L L Lai, D M K AlGobaisi and A Husain, "*Experience of using the neural network approach for identification of MSF desalination plants*" Desalination, 92 (1993) Elsevier Science Publishers B.V., Amesterdam, pp. 323-331
40. Abdulbary A. F., L.L. Lai, D.M.K. Al-Gobaisi, "*Application of Neural Network to Controlling and Modelling of a MSF Desalination Plant*", Thirteen IASTED Int. Conf. MODELLING, IDENTIFICATION AND CONTROL, Feb., 21-23, 1994, Grindelwald, Switzerland.
41. Abdulbary A. F., L L Lai, and D M K Al-Gobaisi, " *Application of a connectioinest Model to controlling a MSF Desalination Plant*", Proceeding of IEEE Conference On Control Applications, Aug. 1994, Glasgow, UK, pp. 1821-1822
42. McCulloch and W. Pitts, "*A logocal calculas of the ideas immanent in nervous activity*", Bulletin of Math. Bio., 5, pp 115-133, 1943.
43. Hebb, *The Organization of Behavior*, Wiley, New York, 1949.
44. Rosenblatt F., " *The Perceptron : A probabilistic model of information storage and organization in the brain*", Psychol. Rev. 65, 386-408, 1958.
45. Widrow B and M.E. Hoff, "*Adaptive Switching Circuits.*", 1960, IRE WESCON Convention Record, part 4, 96-104, New York, August 1960.
46. Minsky and S. Papert, "*Perceptron*", MIT Press, Cambridge MA 1969.
47. Werbos, P. J. 1974. *Beyond regression: New tools for prediction and analysis in the behavioral sciences*. Master thesis, Harvard University.
48. Parker, D.B., *Learning Logic*. Invention Report, S81-64, File 1. Office of Technology Licensing, Stanford University, Stanford, CA.1982.
49. Rumelhart D.E., McClelland J.L. (Eds) 1986. *Parallel Distributed Processing: Exploration in the Microstructure of Cognition, Vol. 1: Foundations*, MIT Press, Cambridge, MA.
50. Grossberg, S. "*Countour enhancement, short-term memory, and consistencies in reverberating neural networks.Studies in Applied Mathematics*: 52:217, 257. 1973
51. Kohonen, T., K. Makisara and T. Saramaki "*Phonotopic maps - insightful representation of the phonological features for speach recognition*", Proceedings of the Seventh International Conference on Pattern Recognition, Motreal, 182-185, 1984.

52. Duda, R.O. and P.E. Hart. "*Pattern Classification and Scene Analysis*", John Wiley & Sons, 1973.
53. Hartigan, J.A. "*Clustering Algorithms*", John Wiley & Sons, 1975.
54. Hammerstorm, D. "*Working with neural networks*", IEEE Spectrum, 46-53, July 1993.
55 Sietsma, J., and Dow, R.J.F., 1991, "*Creating artificial neural networks that generalize*", Neural Networks, 4:67-79
56. Reed, R.D., 1993, *Pruning algorithms: A survey*, IEEE Transaction on neural Network., 4:740-744
57. Shin-Chi Huang and Yif-Fang Huang. "*Bounds on the number of hidden neurons in multilayer perceptrons*", IEEE transaction of neural network, 2(1):47-55, January 1995.
58. Michael A Satori and Panos J. Atsaklis, "*A simple method to drive bounds on the size and to train multilayer neural networks*", IEEE transaction of neural network, 2(4):467-471, July 1991.
59. Hertz J., A. Krogh, and R.G. Palmer 1991. *Introduction to the theory of Neural Computation*, Addison-Wesley Publishing Company, Redwood City, CA.
60. Robert A. Jacobs "*Increased rates of convergence through learning rate adaptation*," Neural Networks, 1:295-307, 1988.
61. Veitch C. and G. Holmes, *A modified quickprop algorithm*", Neural Computation, Vol. 3, 1991, pp. 310-311.
62. Baldi P., and K. Hornik, "*Neural Networks and principal component analysis: Learning from examples and local minima.*", Neural Network, Vol. 2, 1989, pp. 53-58
63. Kuan C.M. and K. Hornik, "*Convergence of learning algorithm with constant learning rate*", IEEE Trans, Neural Networks, Vol. 2, No. 5, 1991, pp. 484 - 490.
64. Marco Gori and Alberto Tesi, "*On the problem of Local Minima in Back propagation*", IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 14, No. 1, January 1992, pp. 76 -86.
65. Brown M. and C. Harris (1994), *Neurofuzzy Adaptive Modelling and Control*, Printice Hall, Englewood Cliffs, NJ., pp. 145.
66. Hornik K., M. Stinchcombe, and H.White, "*Multilayer feed forward networks are universal approximators.*", Neural Networks, 2, pp. 359-366, 1989.
67. Parenti, S. Bogi, and A. Massarani, "*Industrial Application of Real time Neural Networks in Multistage Desalination Plant*", IDA World Congress on "Desalination Water Sciences, Abu Dhabi, Nov. 18-24, Vol. 2, pp. 457-467, 1995
68. Selvaraj Ramasamy, Pradeep B Deshpande, Sanjeev S Tampe, and Bhaskar D Kulkarni "*Identification and Advanced Controls of MSF Desalination Plants with Neural Network*" IDA World Congress on "Desalination and Water Sciences" Abu Dhabi, November 18-24, 1995 Vol VII, pp 36-51
69. Kirsteadter K., *Measurment, Modelling, Identification and Optimized Controller Parameter Settings of MSF-Desalination Units.*, IDA Warld Congress on "Desalination and Water Sciences", Abu Dhabi (1995) Vol VII, 52-81

70. Chen, David S. and Ramesh C. Jain, "*A Robust Back Propagation Learning Algorithm for function Approximation*", IEEE Trans. on Neural Networks, Vol. 5 No. 3 May 1994, 467-479
71. Wong, K.P., and Lau, B.S., "*Fault distance estimation in transient stability assessment: an artificial neural network approach*", Conf. Proc. Fourth International Conference on Expert System Application to Power Systems, Jan., 1993, Melbourne, Australia, pp.1-6.